# NetBeans 学习笔记

## 第一章 NetBeans 集成开发环境

创建第一个 NetBeans 项目

打开 NetBeans，新建项目——〉应用程序——〉项目名称：firstexample,设定存储位置——〉在 main 函数下输入代码：System.out.println(“欢迎使用 NetBeans5.0,这是第一个例子。”);——〉保存——〉运行。

源代码编辑器

尽管可视化开发大大缩短了开发人员的时间，但是代码的编写还是不可被取代的，代码的编写仍然是整个程序的核心部分。一个程序的好坏，很大程度上取决于代码的编写。

NetBeans 提供的代码编辑器十分好用，对于代码的整体性，自动查找错误和修复上面，有了很好的帮助。

## 第二章 Swing 概述

Swing 是 SUN MicroSystem 建立的新一代 GUI 工具包，允许用户进行企业级的开发。Swing 的开发需要许多的包，这里不再一一介绍，当开发的时候，NetBeans 会提醒你添加需要的包，通过修复添加就可以。

下面，我们来进行一个简单得 Swing 程序。再使用了 JDK 以后，我们对 java 语言已经有了一定的认识和了解。虽然 NetBeans 可以使用控件来进行开发，但是必要的代码开发，还是不可少的。在没有习惯以及熟练掌握控件的托拽开发以前，让我们先利用代码来编写一个小的程序。

步骤如下：

建立一个项目，创建一个扩展 Jframe 的类 FirstSwing 用来存放各个组件。在 main()方法中新建一个 FirstSwing 的实例 mySwing。

在 FirstSwing 类中声明各个组件：

```
private JLabel jLabelUserName;
 private JLabel jLabelPassword;
private JTextField jTextFieldUserName;
private JPasswordField jpfPassword;
private JButton jButtonEnter;
private JButton jButtonCancel;
```

将 FirstSwing 的布局管理器设置为 null：

```
this.setLayout(null);//在 FirstSwing 方法中定义。
```

为声明的每个组件创建实例对象：

```
jLabelUserName=new JLabel("用户名：");
        jLabelPassword=new JLabel("密码：");
        jTextFieldUserName=new JTextField();
        jpfPassword=new JPasswordField();
        jButtonEnter=new JButton("确定");
        jButtonCancel=new JButton("取消");
```

设置控件的位置，并且添加到容器中：

```
jLabelUserName.setBounds(10,20,80,20);
        jLabelPassword.setBounds(10,50,80,20);
        jTextFieldUserName.setBounds(100,20,150,20);
```

```
            jpfPassword.setBounds(100,50,150,20);
            jButtonEnter.setBounds(80,90,60,20);
            jButtonCancel.setBounds(160,90,60,20);
            this.add(jLabelUserName);
            this.add(jLabelPassword);
            this.add(jTextFieldUserName);
            this.add(jpfPassword);
            this.add(jButtonEnter);
            this.add(jButtonCancel);
```

设置 FirstSwing 的大小、标题和可见性：

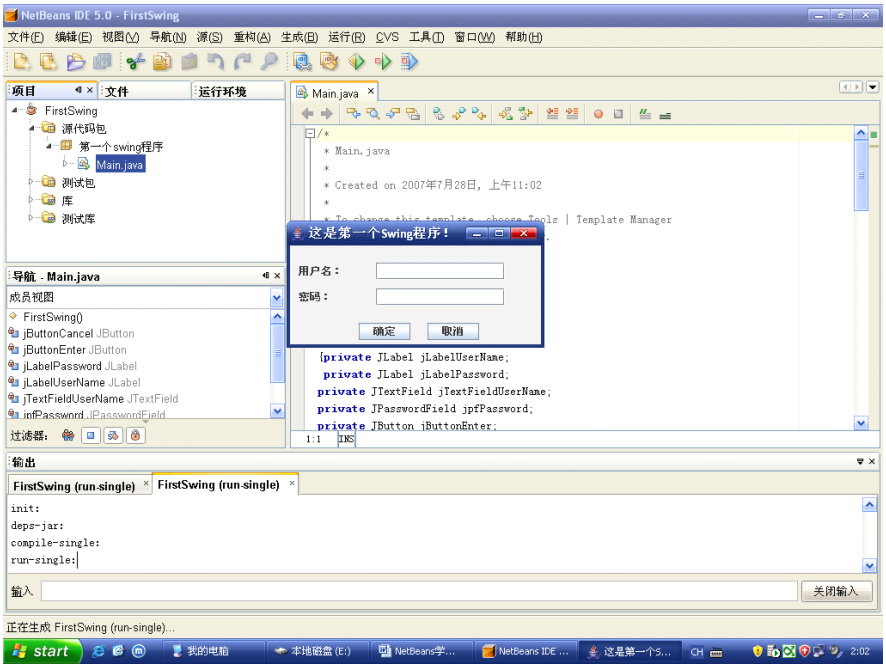
```
this.setBounds(330,250,300,150);
            this.setTitle("这是第一个 Swing 程序！");
            this.setVisible(true);
```

这样，整个程序就完成了，下面给出完整的代码：

```
import java.awt.*;
import javax.swing.*;
  class FirstSwing extends JFrame
{private JLabel jLabelUserName;
  private JLabel jLabelPassword;
private JTextField jTextFieldUserName;
private JPasswordField jpfPassword;
private JButton jButtonEnter;
private JButton jButtonCancel;
          public FirstSwing()
          {
            this.setLayout(null);
            jLabelUserName=new JLabel("用户名：");
            jLabelPassword=new JLabel("密码：");
            jTextFieldUserName=new JTextField();
            jpfPassword=new JPasswordField();
            jButtonEnter=new JButton("确定");
            jButtonCancel=new JButton("取消");
            jLabelUserName.setBounds(10,20,80,20);
            jLabelPassword.setBounds(10,50,80,20);
            jTextFieldUserName.setBounds(100,20,150,20);
            jpfPassword.setBounds(100,50,150,20);
            jButtonEnter.setBounds(80,90,60,20);
            jButtonCancel.setBounds(160,90,60,20);
            this.add(jLabelUserName);
            this.add(jLabelPassword);
            this.add(jTextFieldUserName);
            this.add(jpfPassword);
            this.add(jButtonEnter);
            this.add(jButtonCancel);
```

```
            this.setBounds(330,250,300,150);
            this.setTitle("这是第一个 Swing 程序！");
            this.setVisible(true);
                    }
}
public class Main {
            public static void main(String[] args) {
    FirstSwing mySwing=new FirstSwing();
     }
   }
```

开发后的图像如下：




通过上面的例子，我们已经能够利用 NetBeans 编写一个简单的，具有 GUI 界面的程序了，但是，纯粹的利用代码来编写，十分费力，所以，我们接下来讲的就是如何利用 NetBeans 来开发一个 Swing 应用程序。

步骤如下：

创建项目——〉将工程的名字修改为 addstudentinfo，并且设置包的位置，一般设置位 org.netbeans,选中左方窗口的源代码包下的 netbeans 节点，鼠标右击，新建 JFrame，并且修改名字为 AddStudentFrame，然后确定。打开 GUI 设计器。

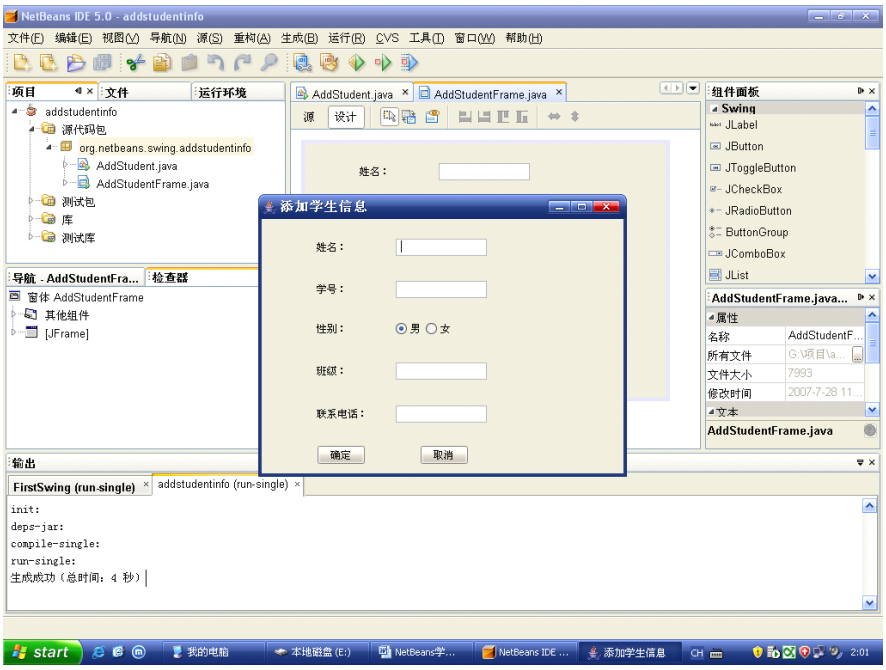
向窗体上添加一个 Jlabel,鼠标右击——〉改变变量名称：jLabelStudentName,属性：text ：姓名。然后依次再添加 4 个 Jlabel，名称分别为 jLabelStudentID(学号：)，jLabelGender（性别：）,jLabelGrade(班级：),jLabelPhoneNum(联系电话：)。

添加四个 JtextField,分别为 jTextFieldStudentID,jTextFieldGrade,jTextFieldPhoneNum,还有 jTextFieldName。全部设置为空。

两个 JradioButton,JradioButtonMale(男)，JRadioButtonFemale(女);

两个 JButton，JbuttonEnter(确定)，JbuttonCancel(取消);

运行后的图像如下：




第三章 Swing/JFC 的事件模型基础

任何支持 GUI 的操作环境都会不断地监听事件。当按下键盘或者单击鼠标的时候，便会触发一个相应的事件，操作环境把这些事件报告给正在运行的程序，接着程序决定如何响应这些事件。在 java 中，开发人员能够控制各种事件，可以把迅速响应事件的对象指派成一个事件监听器。

下面介绍一下常用事件对象及其作用。

AWTEvent：所有 AWT 事件的根事件类。

ConnectionEvent：封装与连接有关信息的事件。

ChangeEvent ：封装状态改变信息的事件。

ListSelectionEvent：表示当前选择中更改的特征事件。

CaretEvent：封装文本区中的光标改变信息的事件。

ListDataEvent：定义一个封装列表更改的事件。

HyperlinkEvent：封装与超文本连接有关的事件。

TableColumnModelEvent：某一个表的列模型已发生改变。

TableModelEvent：通知侦听器某一个表模型已发生改变。

TreeExpansionEvent：用于表示树中的单个路径的事件。

TreeModelEvent：封装描述树模型更改的信息。

TreeSelectionEvent：描述当前选择的更改事件。

上面所述的，只是一些常用的事件对象和作用，当遇到那些不常用的时，应该第一时间查阅 JDK。

在了解了事件处理的工作机制以后，现在深入讨论 java 中的事件处理。Java 中的事件处理是面向对象的，所有事件都是从 java.util 包中的 EventObject 类扩展来的。EventObject 有一个子类 AWTEvent，它是所有 AWT 事件的父类。

对于 java 开发人员来说，有些 AWT 事件在实际中应用不多。我们所讲述的，只是在实际中会被传到舰艇气得 AWT 事件类型。

常用的事件类型和接口

| 事件类型 | 接口名称 | 接口中声明的方法 | 作用 |
| --- | --- | --- | --- |
| ActionEvent | ActionEventListener | actionPerformed(ActionEvent e) | 监听组件的某个动作 |
| AdjustmentEvent | AdjustmentEventListener | adjustmentValueChanged(AdjustmentEvent e) | 监听调整事件 |
| ComponentEvent | ComponentListener | componentHidden(ComponentEvent e) | 监听组件隐藏事件 |
| 同上 | 同上 | ……Moved…… | 移动事件 |
| 同上 | 同上 | ……Resized…… | 调整尺寸 |
| 同上 | 同上 | ……Shown…… | 显示事件 |
| ContainerEvent | ContainerEventListener | componentAdded(ContainerEvent e) | 监听容器添加组件事件 |
| 同上 | 同上 | ……Removed…… | 移除事件 |
| FocusEvent | FocusListener | focusGained(FocusEvent e) | 监听键盘获得焦点事件 |
| 同上 | 同上 | ……Lost…… | 失去焦点 |
| HierarchyEvent | HierarchyBoundsListener | ancestorMoved(HierarchyEvent e) | 监听父窗口的移动事件 |
| 同上 | 同上 | ……Resized…… | 调整尺寸 |
| InoputMethodEvent | InputMethodListener | inputMethodTextChanged(InoputMethodEven) | 监听输入方式改变事件 |
| ItemEvent | ItemListener | itemStateChanged(ItemEvent e) | 是否选种状态的改变 |
| KeyEvent | KeyListener | keyPressed(KeyEvent e) | 监听键盘按下事件 |
| 同上 | 同上 | ……Released…… | 释放事件 |
| 同上 | 同上 | ……Typed…… | 单击事件 |
| MouseEvent | MouseListener | mouseClicked(MouseEvent e) | 监听鼠标的单击事件 |
| 同上 | 同上 | ……Entered…… | 移入 |
| 同上 | 同上 | ……Exited…… | 移出 |
| 同上 | 同上 | ……Pressed…… | 按下 |
| 同上 | 同上 | ……Released…… | 释放 |
| 同上 | MouseMotionListener | mouseDragged(Event e) | 鼠标在组件内的托拽事件 |
| 同上 | 同上 | ……WheelMoved…… | 同上 |

以上就是几种最常用的事件及监听器，此外，还有窗口事件（WindowEvent）。

## 第四章 Swing 常用基本控件

JLabe 类
JLabel myjLabel=new JLabel();
构造器
JLabel():创建无图像并且标题为空的 JLabel

JLabel(Icon image):  创建具有指定图像的 JLabel

JLabel(String text):创建具有指定文本的 JLabe

JLabel(String text,Icon image,int horizontalAlignment()):具有指定文本，图像，水平对齐德 JLabel

JLabel(String text,int horizontalAlignment()):具有指定文本和水平对齐的 JLabel

常用方法

getText():获取该标签所显示的文本字符串。

setText(String text):设置字符串。

getIcon():获取该标签所显示的图像。

setIcon(Icon icon):定义此组件将要现实的图标。

getVerticalAlignment():获取标签内容沿 y 轴的对齐方式。

setVerticalAlignment(int alignment):设置。

getHorizontalAlignment():获取，x 轴。

setHorizontalAlignment(int alignment):设置。

getLabelFor():获取将标签添加到的组件。

setLabelFor(Component c):设置将标签添加到的组件。

在 NetBeans 中使用 JLabel

项目——〉新建 Jframe(DemoJlabelFrame)——〉添加 JLabel(jLabelImage),text:演示图片。单击 Icon 属性右侧的按钮，添加图片。——〉右击，选择“事件/Mouse/mouseClicked”，然后添加如下代码：

This.jLabelImage.setVerticalTextPosition(SwingConstants.TOP);

运行即可。运行图象如下：


JButton 介绍

Swing 中最简单的按钮类型，可以包含文本或图标，能够响应单击事件。

创建十分简单：

JButton myButton=new JButton(“确定”);

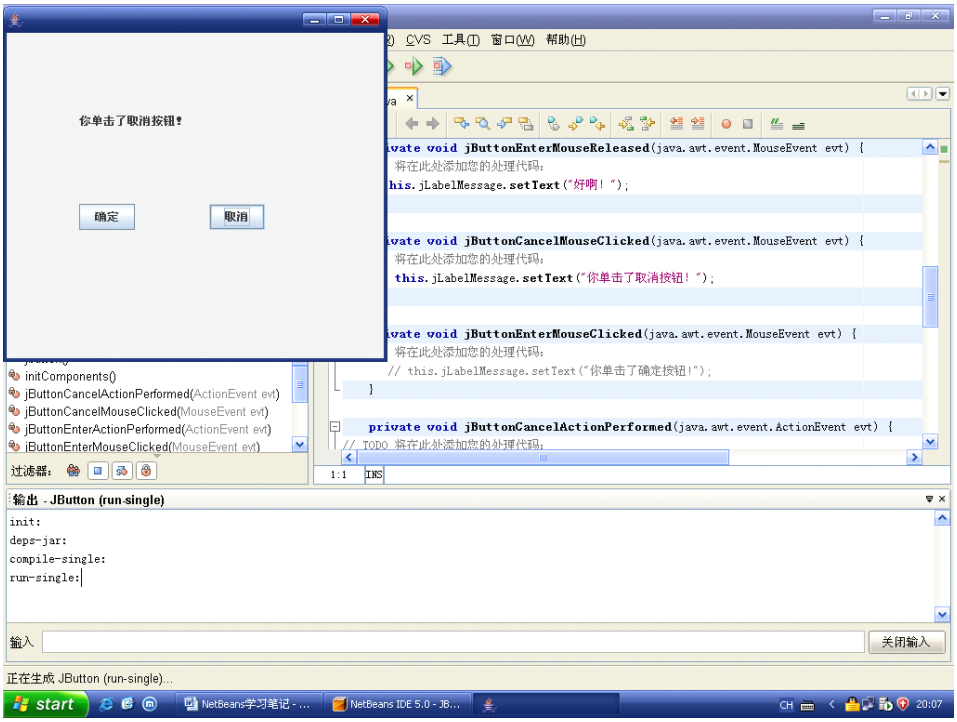
构造器：
JButton():创建没有文本或图标的按钮。
JButton(String s):有文本。
JButton(Icon icon):有图标。
JButton(String s,Icon icon)

常用方法
addActionListener()
removeActionListener(ActionListener l)
getAction()
getModel()
setModel(ButtonModel newModel)
getMargin()
setMargin(Insets m)
getSelectedIcon()
setSelectedIcon(Icon selectedIcon)

在 NetBeans 中 JButton 的使用

新建 JFrame(DemoJButtonFrame)——〉添加 2 个 JButton,1 个 JLabel,分别修改变量名称为 jButtonEnter,jButtonCancel,jLabelMessage。Test 属性分别为：确定，取消，用户没有按下按钮。分别为两个按钮添加 ActionEvent 事件。

jButtonEnter:
This.jLabelMessage.setText(“你单击了确定按钮”);
jButtonCancel:
This.jLabelMessage.setText(“你单击了取消按钮”);
效果图：




Swing 文本框

JtextField

用于输入单行的文字。创建很简单：

JtextField jTextFieldOne=new JtextField(“你好”);

构造器：

JTextField()

JTextField(Document doc,String text,int columns):给定存储类型和列数，还有相应的文本。

JTextField(int columns)

JTextField(String text)

JTextField(String text,int columns)

常用方法：

setDocument(Document doc)

createDefaultModel()

getColumns()

setColumns(int columns)

setFont(Font f)

addActionListener(ActionListener l)

removeActionListener(ActionListener l)

getText()

setText(String t)

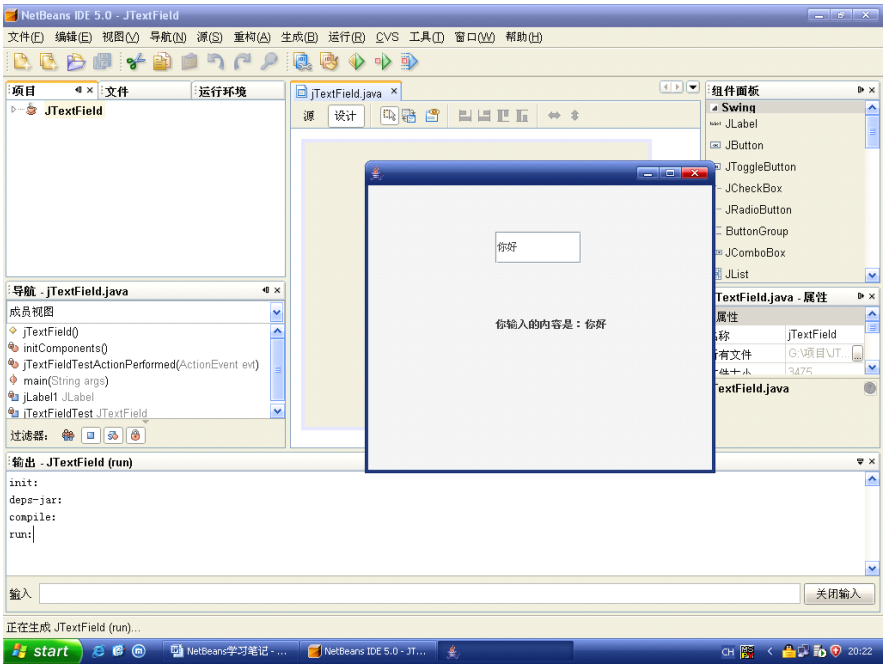
在 NetBeans 中使用

添加 JTextField(jTextFieldTest),JLabel(jLabelMessage)

在 jTextFieldTest 的 ActionEvent 下输入代码：

```
String getMessage=jTextFieldTest.getText();
If(getMessage.equals(“”))
{
    this.jLabelMessage.setText(“你没有输入任何内容”);
}
Else
{
    this.jLabelMessage.setText(“你输入的内容是：”+ getMessage)
}
```

效果图：




Swing 文本区

JTextArea 是一种非常重要的文本输入控件。能够显示多行文本。

构造器：

JTextArea()

JTextArea(String text)

JTextArea(int rows,int columns)

JTextArea(String text,int rows,int columns)

JTextArea(Document doc)

JTextArea 本身不具有滚动功能，如要实现该功能，需要将其添加到 JScrollPane 中，下面是代码：

```
JScrollPane    myScrollPane=new JScrollPane(myTextArea);
```

常用方法：

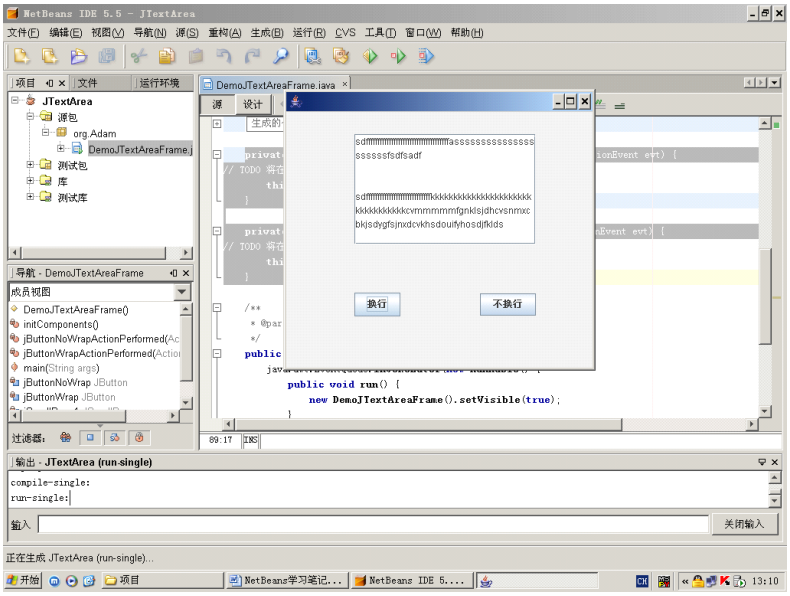
setLineWrap(Boolean wrap)

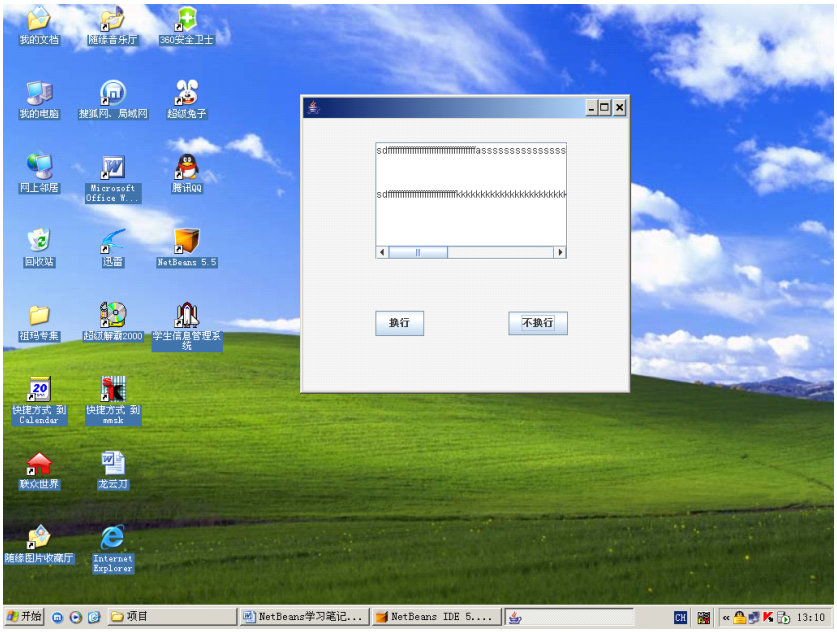
getLineWrap()
insert(String str,int pos)
append(String str)
getRows()
setRows(int rows)
getColumns()
setColumns(int columns)
还有其他的一些方法，可以查阅 API，在练习的过程中，也需要一点点体会。

在 NetBeans 中使用

代码和运行图如下：

```
private void jButtonNoWrapActionPerformed(java.awt.event.ActionEvent evt) {
// TODO  将在此处添加您的处理代码：
            this.jTextAreaTest.setLineWrap(false);
        }

        private void jButtonWrapActionPerformed(java.awt.event.ActionEvent evt) {
// TODO  将在此处添加您的处理代码：
            this.jTextAreaTest.setLineWrap(true);
        }
```






Swing 单选按钮（JRadioButton）与复选框(JCheckBox)

JCheckBox
构造器
JCheckBox(Icon icon)
JCheckBox(Icon icon,Boolean selected)
JCheckBox(String text)
JCheckBox(String text,Boolean selected)
JCheckBox(String text,Icon icon)

常用方法：
isSelected()
setSelected(Boolean b)

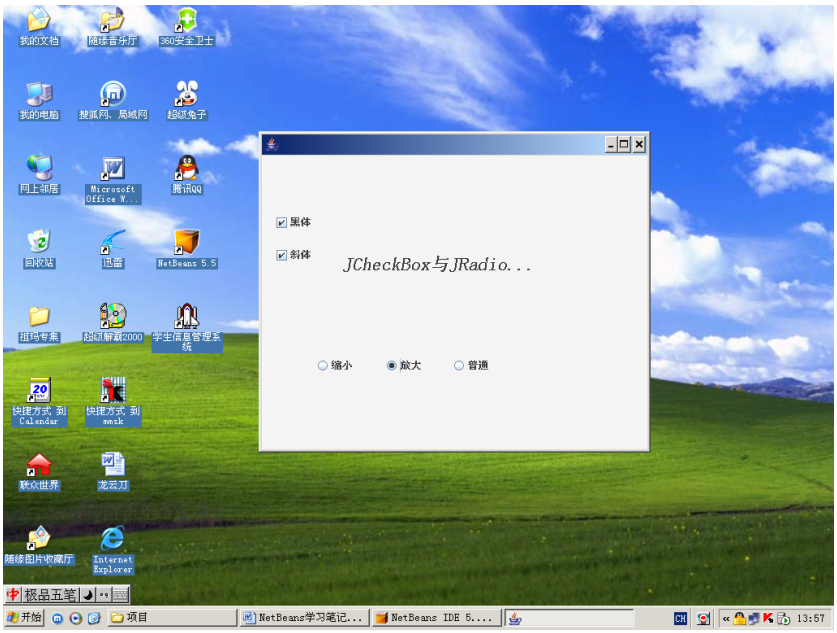
JRadioButton
构造器
JRadioButton(Icon icon)
JRadioButton(Icon icon,Boolean selected)
JRadioButton(String text)
JRadioButton(String text,Boolean selected)
JRadioButton(String text,Icon icon)

因为单选按钮在某一时刻，一组单选按钮中只能有一个被选中，所以需要添加 ButtonGroup， 然后将相应的一组单选按钮添加到 ButtonGroup 中。这样才可以满足在一个时刻，只有一个被选中。

在 NetBeans 中使用
代码和运行图如下

```
//定义一些方法和成员变量
private int style = java.awt.Font.PLAIN;

    private int fontSize = 18;

    private void setStyle(int style) {
        this.style=this.style+style;
    }

    private int getStyle() {
        return this.style;
    }

    private void setFontSize(int fontSize) {
        this.fontSize=fontSize;
    }

    private int getFontSize() {
        return this.fontSize;
    }

    private void setLableFont() {
        this.jLabel1.setFont(new java.awt.Font("宋体",this.getStyle(),this.getFontSize()));
    }

//事件中的代码
    private void jCheckBox2ItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
        if(this.jCheckBox2.isSelected()){
            this.setStyle(java.awt.Font.ITALIC);
        }else{
            this.setStyle(-java.awt.Font.ITALIC);
        }this.setLableFont();
    }

    private void jRadioButton3ItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
        this.setFontSize(18);
        this.setLableFont();
    }
```

```
    private void jRadioButton2ItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
        this.setFontSize(22);
        this.setLableFont();
    }

    private void jRadioButton1ItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
        this.setFontSize(14);
        this.setLableFont();
    }

    private void jCheckBox1ItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
        if(this.jCheckBox1.isSelected()){
            this.setStyle(java.awt.Font.BOLD);
        }else{
            this.setStyle(-java.awt.Font.BOLD);
        }this.setLableFont();
    }
```



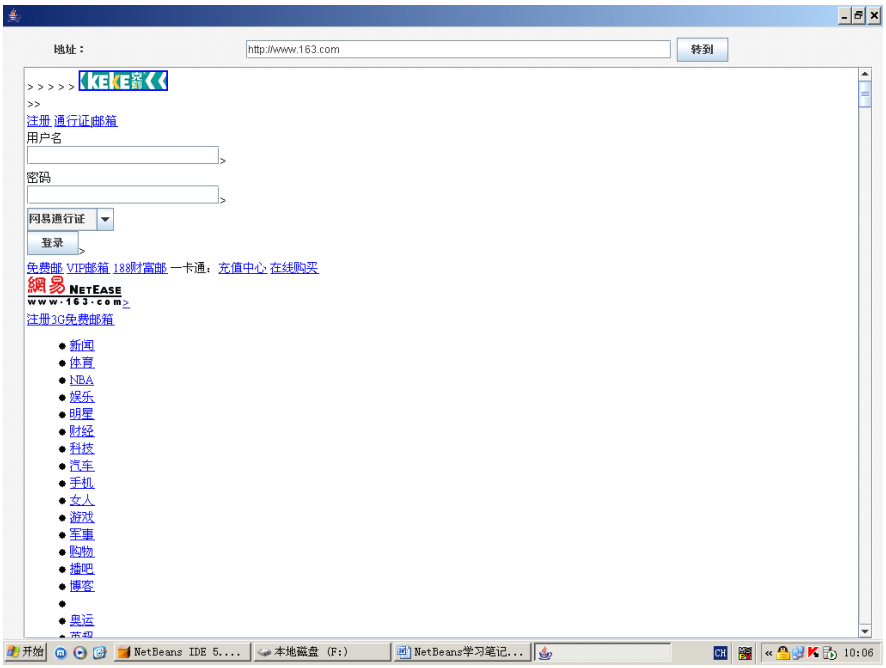

编辑器面板（JEditorPane）

他是 JTextComPonent 的子类，可以显示多种格式内容，例如 HTML、RTF，可以做简单的 HTML 浏览器的工具。

构造器：
JEditorPane()
JEditorPane(java.net.URL initialPage)
JEditorPane(String url)
JEditorPane(String type,String text)

常用方法：
addHyperlinkListener(HyperlinkListener listener)
getHyperlinkListeners()
getPage()
setPage(String url)
setPage(java.net.URL page)
getStream(java.net.UAL page)
在 NetBeans 中使用
代码和运行图如下

```
private void jButtonGoActionPerformed(java.awt.event.ActionEvent evt) {     //按钮事件
// TODO 将在此处添加您的处理代码：
        String myURL=this.jTextFieldAddress.getText().trim();

        try{
            if(myURL != null &&! myURL.equals(""))
                jEditorPaneHTML.setPage(myURL);
        }catch(java.io.IOException e){
            e.printStackTrace();
        }

    }

    private void jEditorPaneHTMLHyperlinkUpdate(javax.swing.event.HyperlinkEvent evt)
{                                                           //显示页面
// TODO 将在此处添加您的处理代码：
        try{

if(evt.getEventType()==javax.swing.event.HyperlinkEvent.EventType.ACTIVATED)
                jEditorPaneHTML.setPage(evt.getURL());
        }catch(java.io.IOException e){
            e.printStackTrace();
        }
    }
```




第五章

5.1 Swing 列表

JList

创建代码：

String [] fruit={“apple”,”pear”,”orange”,”banana”}

JList myJList=new JList(fruit);

构造器：

JList()

JList(Object [] listData)

JList(ListModel dataModel)

JList(Vector listData)

常见方法：

getSelecttionForegroud()

setSelectionForeGroud(Color selectionForegroud)

getSelectionBackgroud()

setSelectionBackgroud(Color selectionBackgroud)

getModel()

setModel(ListModel model)

setListData(Object[] listData)

setListData(Vector listData)

setSelectionMode(int selectionMode)

getSelectionMode()

getMaxSelectionIndex()

getMinSelectionIndex()

isSelectedeIndex(int index)

isSelectionEmpty()

clearSelection()

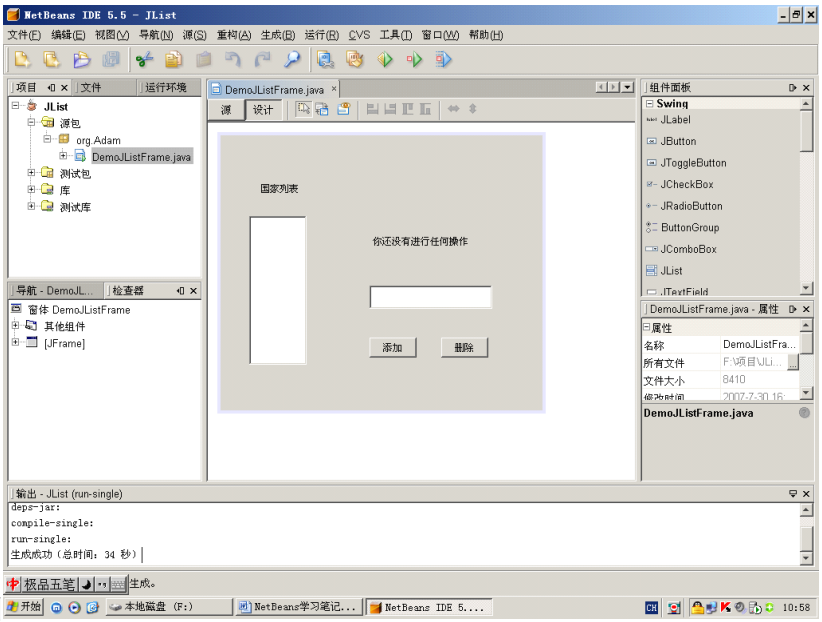
getSelectedValues()

在 NetBeans 中使用

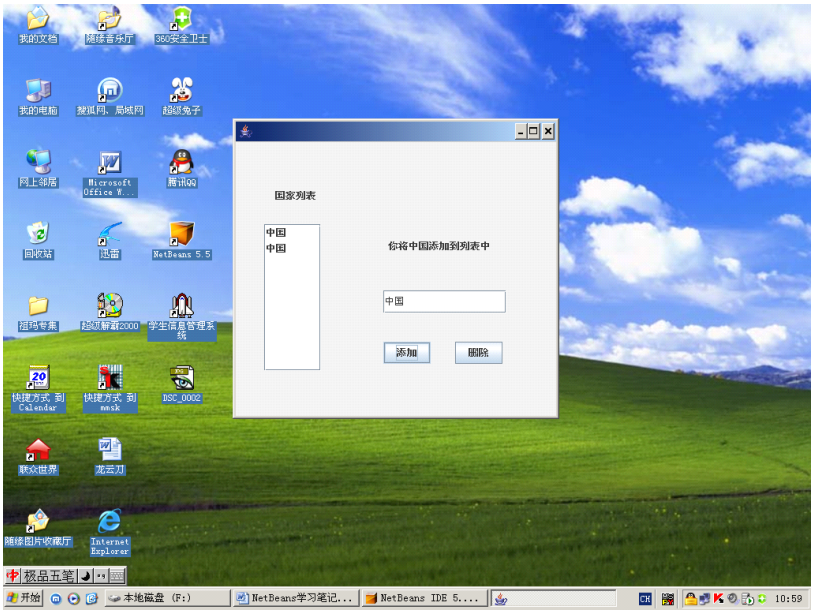
代码和运行图如下

```
private void jTextFieldCountryNameActionPerformed(java.awt.event.ActionEvent evt) {
// TODO 将在此处添加您的处理代码：
        String countryName=this.jTextFieldCountryName.getText().trim();
        if(countryName !=null &&! countryName.equals("")){
            this.imageVector.add(countryName);
            this.jLabelActionMessage.setText("你将"+countryName+"添加到列表中");
            this.jListCountryList.setListData(this.imageVector);

        }else{this.jLabelActionMessage.setText("请先输入内容再添加");
        }
    }

    private void jButtonDeleteActionPerformed(java.awt.event.ActionEvent evt) {
// TODO 将在此处添加您的处理代码：
        String countryName=(String)this.jListCountryList.getSelectedValue();
        if(countryName !=null){
            this.imageVector.remove(countryName);
            this.jListCountryList.setListData(imageVector);
            this.jLabelActionMessage.setText("你将"+countryName+"从列表中删除");
        }else{
            this.jLabelActionMessage.setText("请先选择一个选项");
        }
    }

    private void jButtonAddActionPerformed(java.awt.event.ActionEvent evt) {
// TODO 将在此处添加您的处理代码：
        String countryName=this.jTextFieldCountryName.getText().trim();
        if(countryName !=null &&! countryName.equals("")){
            this.imageVector.add(countryName);
            this.jLabelActionMessage.setText("你将"+countryName+"添加到列表中");
            this.jListCountryList.setListData(this.imageVector);

        }else{
            this.jLabelActionMessage.setText("请先输入内容再添加");
        }
    }

    private void jListCountryListValueChanged(javax.swing.event.ListSelectionEvent evt) {
// TODO 将在此处添加您的处理代码：
        String choosedCountry=(String)this.jListCountryList.getSelectedValue();
```

```
        this.jLabelActionMessage.setText("你从列表中选择"+choosedCountry+"选项");
}
```




这是设计图




Swing 组合框(JComboBox)

构造器：

JComboBox()

JComboBox(Object[] items)

JComboBox(Vector items)

JComboBox(ComboBoxModel model)

常用方法：

setModel(ComboBoxModel aModel)

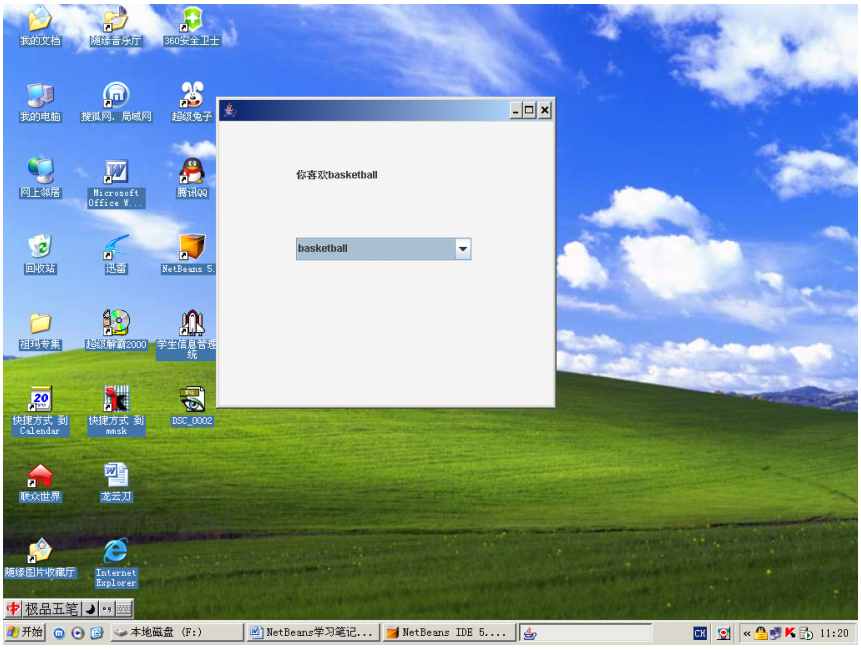
getModel()

setSelectedItem(Object anObject)

getSelectedItem()
setSelectedIndex(int anIndex)
getSelectedIndex()
addItem(Object anObject)
removeItem(Object anObject)
removeItemAt(int anIndex)
removeAllItems()
showPopup()
hidePopup()
setPopupVisible(Boolean v)
isPopupVisible()
getItemCount()
getItemAt(int index)

在 NetBeans 中使用
代码和运行图如下

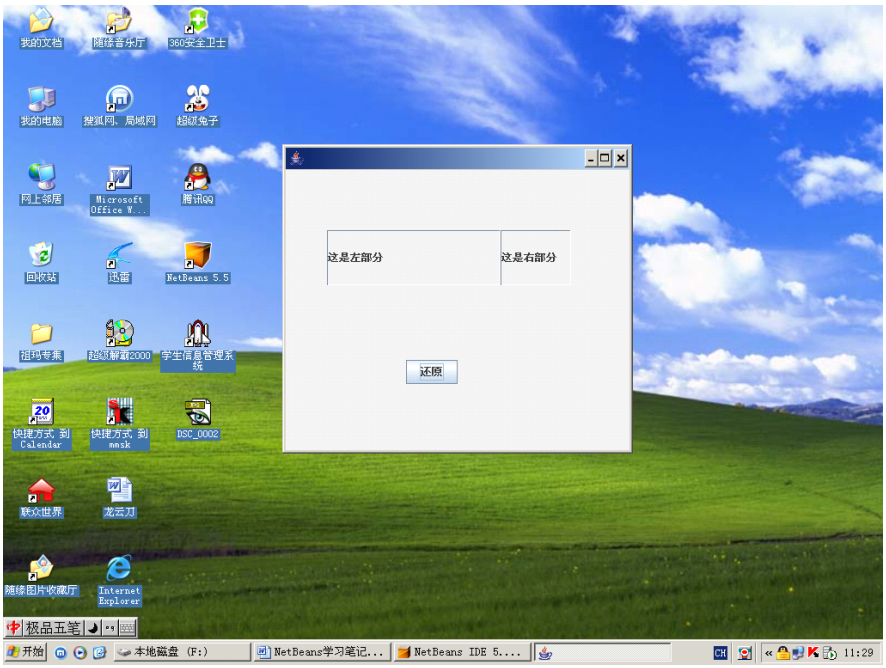
```
//先添加一个 JComboBox，然后选择 model 属性，删除原来的项，添加新的项。
private void jComboBoxLoverItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO  将在此处添加您的处理代码：
            String favor=(String)this.jComboBoxLover.getSelectedItem();
             this.jLabel1.setText("你喜欢"+favor);

    }
```

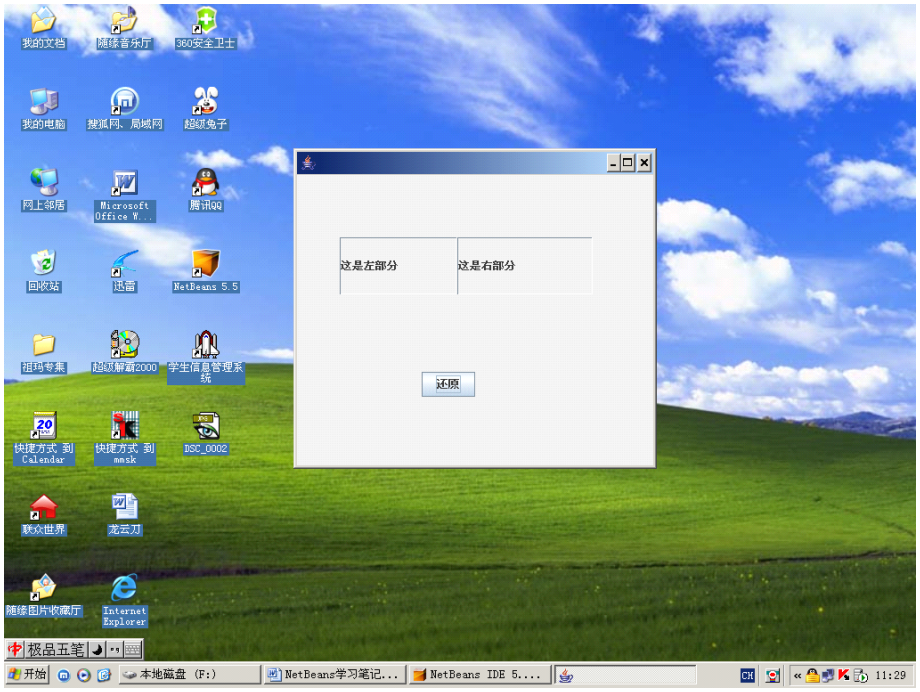



Swing 分割窗口（JSplitPane）
用于将一个窗口分割成两个部分。
构造器
JSplitPane()

JSplitPane(int newOrientation)
JSplitPane( int newOrientation,Boolean newContinuousLayout)

常用方法：
setDividerSize(int newSize)
getDividerSize()
setLeftComponent(Component comp)
Component getLeftComponent()



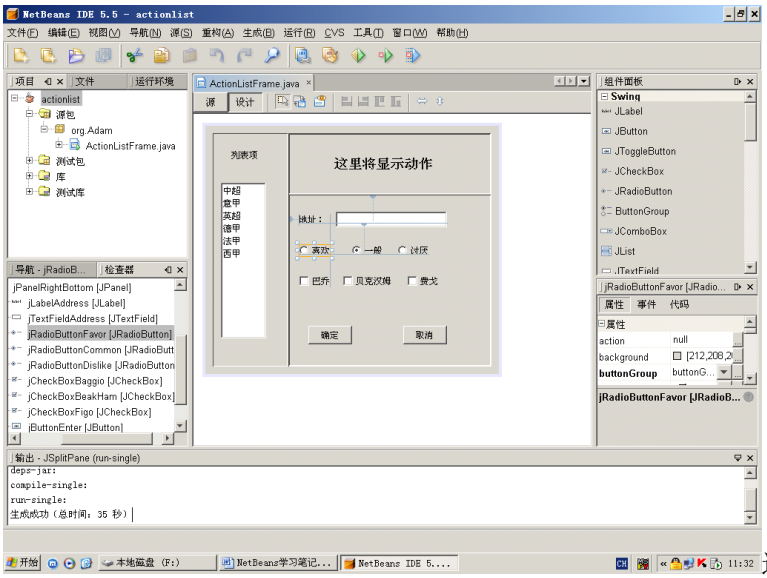




实战：使用 NetBeans 构建 Swing 的综合例程
代码如下：

```
private void jButtonCancelActionPerformed(java.awt.event.ActionEvent evt) {
// TODO 将在此处添加您的处理代码：
        this.jLabelMessage.setText("你按下了取消按钮");
    }

    private void jButtonEnterActionPerformed(java.awt.event.ActionEvent evt) {
// TODO 将在此处添加您的处理代码：
        this.jLabelMessage.setText("你按下了确定按钮");
    }

    private void jCheckBoxFigoItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
        if(this.jCheckBoxBaggio.isSelected()){
            this.jLabelMessage.setText("你开始喜欢费戈了");
        }else{
            this.jLabelMessage.setText("你不喜欢费戈了");
        }
    }

    private void jCheckBoxBeakHamItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
        if(this.jCheckBoxBaggio.isSelected()){
            this.jLabelMessage.setText("你开始喜欢贝克汉姆了");
        }else{
            this.jLabelMessage.setText("你不喜欢贝克汉姆了");
        }
    }

    private void jCheckBoxBaggioItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
        if(this.jCheckBoxBaggio.isSelected()){
            this.jLabelMessage.setText("你开始喜欢巴乔了");
        }else{
            this.jLabelMessage.setText("你不喜欢巴乔了");
        }
    }

    private void jRadioButtonDislikeItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
         this.jLabelMessage.setText("你选择了讨厌按钮");
    }

    private void jRadioButtonCommonItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
```

```
            this.jLabelMessage.setText("你选择了一般按钮");
        }

        private void jRadioButtonFavorItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
            this.jLabelMessage.setText("你选择了喜欢按钮");
        }

        private void jTextFieldAddressActionPerformed(java.awt.event.ActionEvent evt) {
// TODO 将在此处添加您的处理代码：
            String address=this.jTextFieldAddress.getText();
            if(address !=null &&! address.equals("")){
                this.jLabelMessage.setText("你输入了"+address);
            }else{
                this.jLabelMessage.setText("你没有输入任何内容");
            }
        }

        private void jListSportValueChanged(javax.swing.event.ListSelectionEvent evt) {
// TODO 将在此处添加您的处理代码：
            String sport=(String)this.jListSport.getSelectedValue();
            this.jLabelMessage.setText("你选择了"+sport);
        }
```

设计图如下：




运行图如下：


第 6 章

Swing 框架——JFrame

创建代码：

JLabel myJLabel=new JLabel(“使用 JFrame”);

JFrame myJFrame=new JFrame();

myJFrame.add(myJLabel);

构造器：

JFrame()

JFrame(String title)

JFrame(GraphicsConfiguration gc)

JFrame(String tit;e,GraphicsConfiguration gc)

常用方法：

setDefaultCloseOperation(int operation)

getDefaultCloseOperation()

updata(Graphics g)

setJMenuBar(JMenuBar menubar)

getJMenuBar()

setTitle(String title)

dispose()

setResizable(boolean b)

setIconImage(Image images)

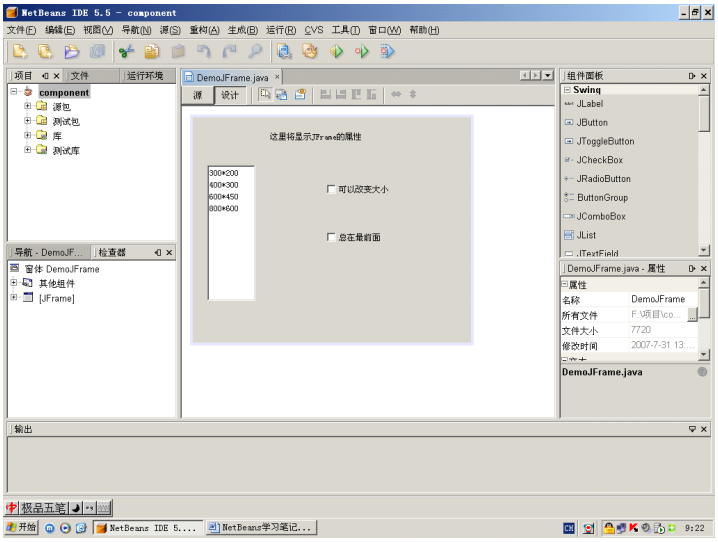
setVislble(Boolean b)

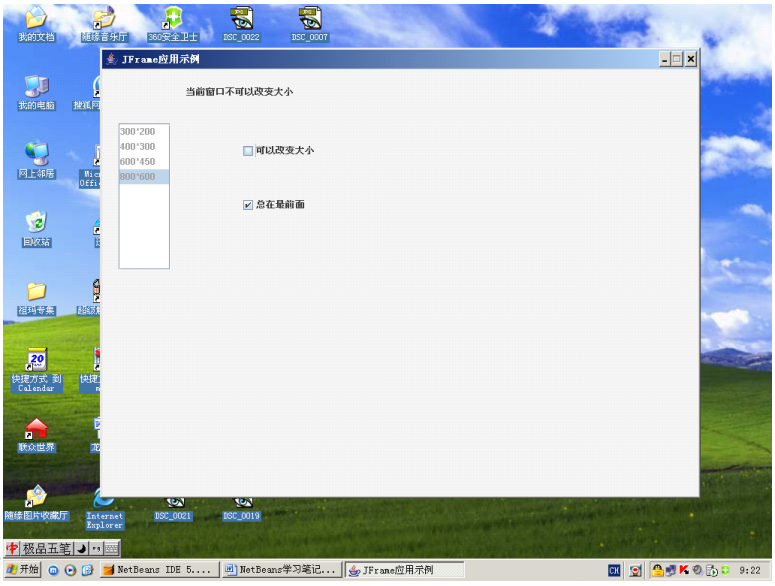
setAlwaysOnTop(Boolean alwaysOnTop)

isAlwaysOnTop()

在 NetBeans 中使用

代码和运行图如下






运行效果图

```
        private void jListSizeValueChanged(javax.swing.event.ListSelectionEvent evt) {
// TODO  将在此处添加您的处理代码：
            int frameSize=this.jListSize.getSelectedIndex();
            switch(frameSize){
                case 0:
                    this.setSize(300,200);
                    this.jLabelMessage.setText("当前窗口大小为 300*200");
                    break;
                case 1:
                    this.setSize(400,300);
                    this.jLabelMessage.setText("当前窗口大小为 400*300");
                    break;
                case 2:
```

```
                this.setSize(650,400);
                this.jLabelMessage.setText("当前窗口大小为 650*400");
                break;
            case 3:
                this.setSize(800,600);
                this.jLabelMessage.setText("当前窗口大小为 800*600");
                break;
        }

    }

    private void jCheckBoxOnTopItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
        if(this.jCheckBoxOnTop.isSelected()){
            this.setAlwaysOnTop(true);
            this.jLabelMessage.setText("当前窗口总位于最前面");
        }else{
            this.setAlwaysOnTop(false);
            this.jLabelMessage.setText("当前窗口不是总位于最前面");
        }
    }

    private void jCheckBoxSizeItemStateChanged(java.awt.event.ItemEvent evt) {
// TODO 将在此处添加您的处理代码：
        if(this.jCheckBoxSize.isSelected()){
            this.setResizable(true);
            this.jListSize.setEnabled(true);
            this.jLabelMessage.setText("当前窗口可以改变大小");
        }else{
            this.setResizable(false);
            this.jListSize.setEnabled(false);
            this.jLabelMessage.setText("当前窗口不可以改变大小");
        }
    }
```

Jpanel 的使用

Jpanel 是 swing 包中一个非常重要的控件，在程序的开发中，我们经常会使用到 Jpanel，这样可以大大的规范我们的布局，而却有些模式，必须要用到 Jpanel。

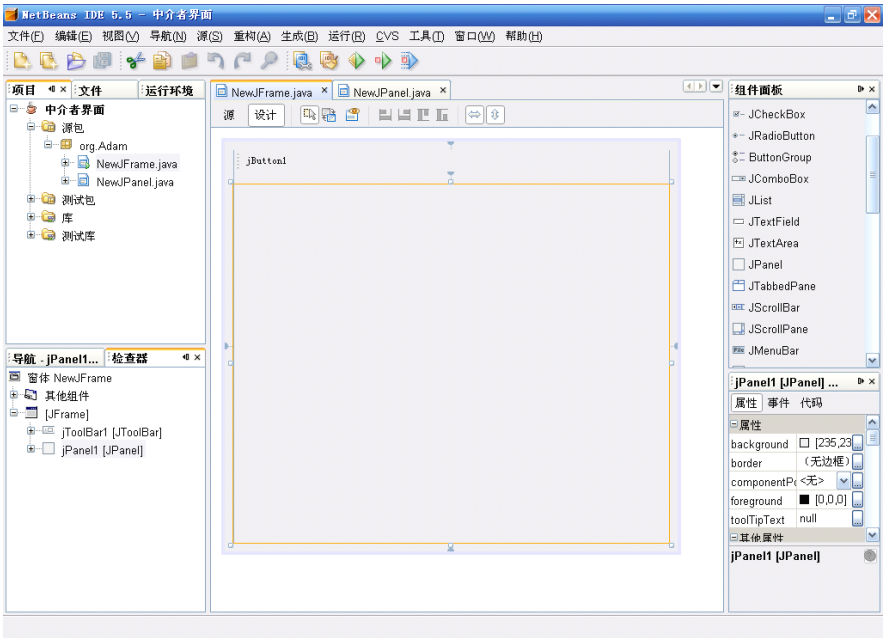
Jpanel 与 Jframe 十分相像，并却可以添加在 Jframe 上。在他的 Border 属性中，我们可以选择 TitledBorder，然后可以添加标题。这样我们的界面设计就会十分合理。

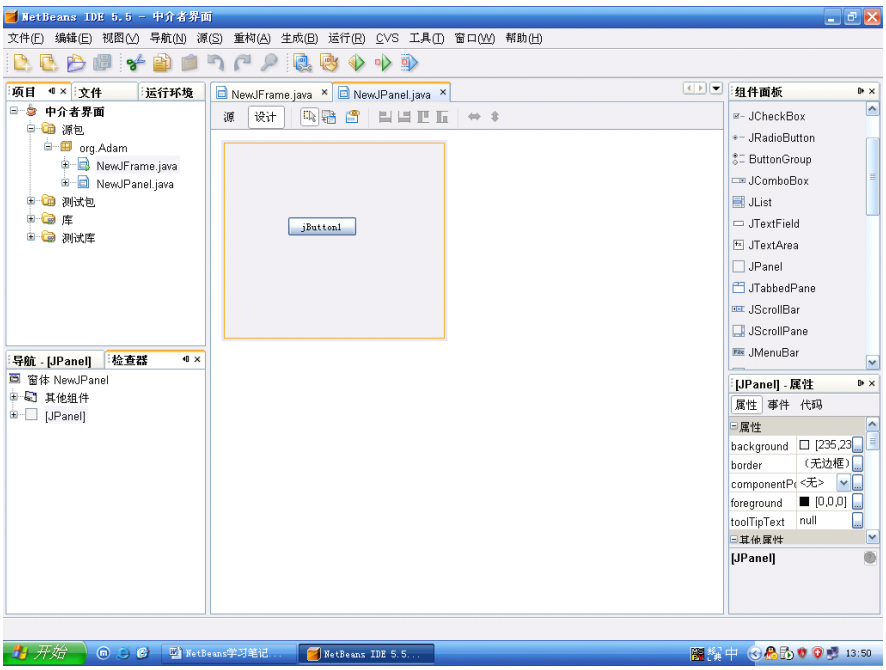
Jpanel 的构造器，与 Jframe 基本上大致相同，在此不必多说。有关的问题，都可以在 JDK 上面查到。

下面，我们来看一个例子。

中介者的界面使用：

创建一个 Jframe 和一个 Jpanel ,图形如下：




在 Jframe 上添加一个 jPanel，并去设置布局为 BorderLayout.

然后添加如下代码：

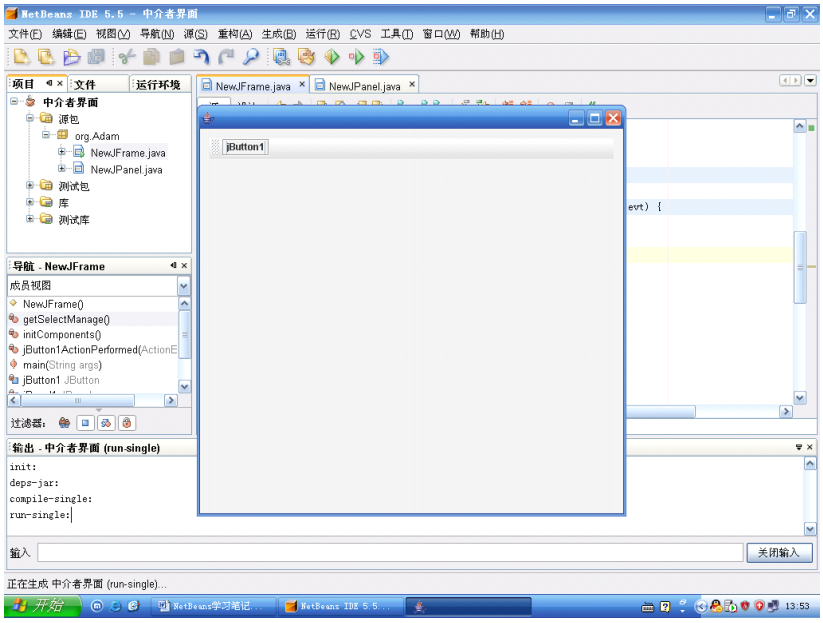
```
NewJPanel selectManage=null;//在类中声明
```

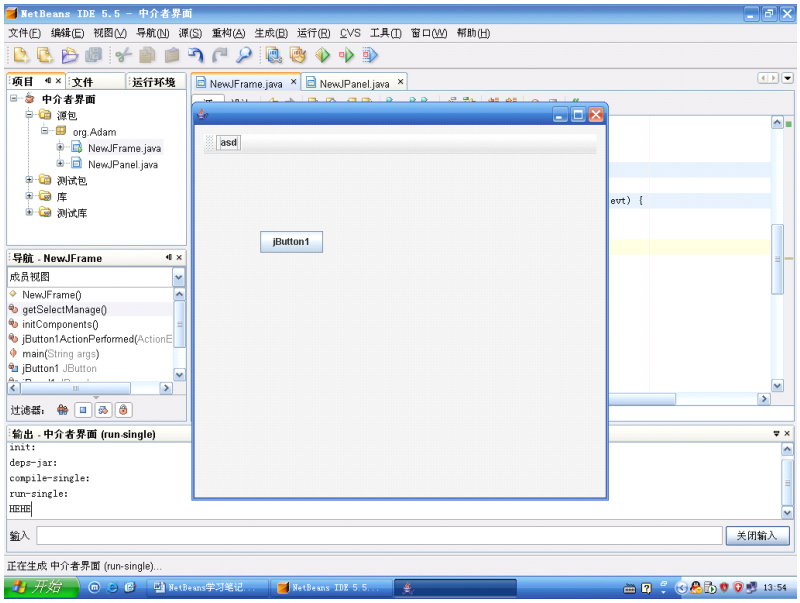
添加按钮事件：

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO  将在此处添加您的处理代码：
        if(selectManage==null){
            selectManage=new NewJPanel();
         }
        this.jPanel1.add(selectManage,BorderLayout.CENTER);
```

```
        this.repaint();
                this.jButton1.setText("asd");       }
```

运行效果图：




关于图标问题

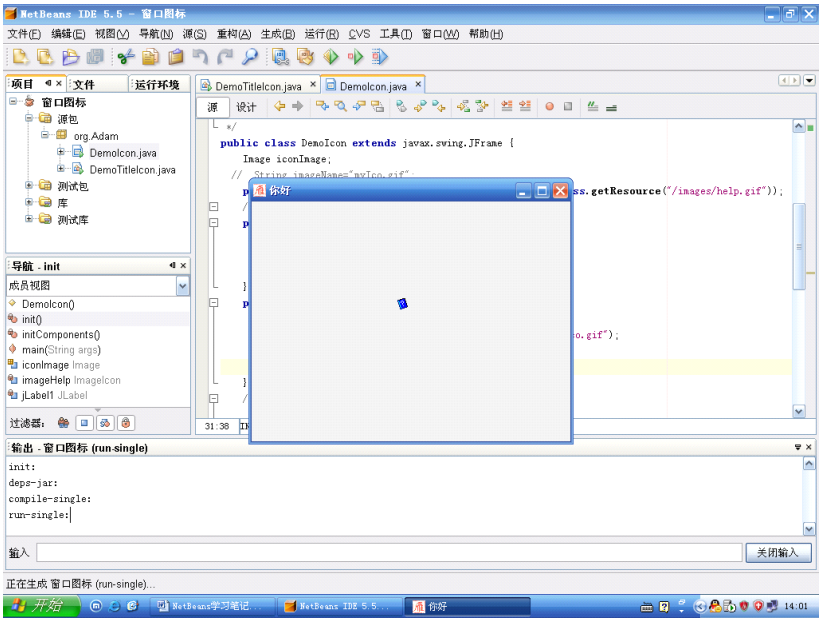
我们使用 NetBeans 做好的页面，都已经有了 NetBeans 所提供给我们的图标，但是真正在完成一个项目的时候，我们都要使用一个自己所做好的图标，满足用户的需求。所以，我们就来研究一下，怎样自己设置图标。

首先，创建一个窗体，并在该项目文件夹下，放一个名字为 myIco.gif 的图片。

添加代码：

```
Image iconImage;//在类中声明
在该类下的构造函数下声明
iconImage=Toolkit.getDefaultToolkit().getImage("myIco.gif");
this.setIconImage(iconImage);
```

效果图：



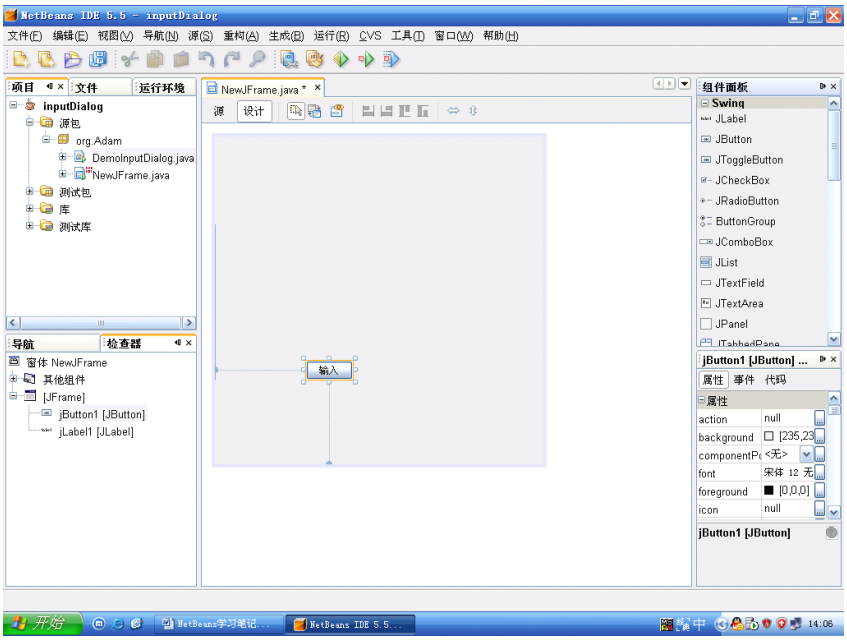

大家可以看一下，窗口的图标已经改变了。

## 输入对话框的使用

在我们设计程序的时候，很多时候我们需要用到对话框。对话框有很多种类，而且应用也十分广泛。在此，我们先讲第一种，那就是输入对话框。

输入对话框的创建十分简单，下面我们看一个例子：

设计图

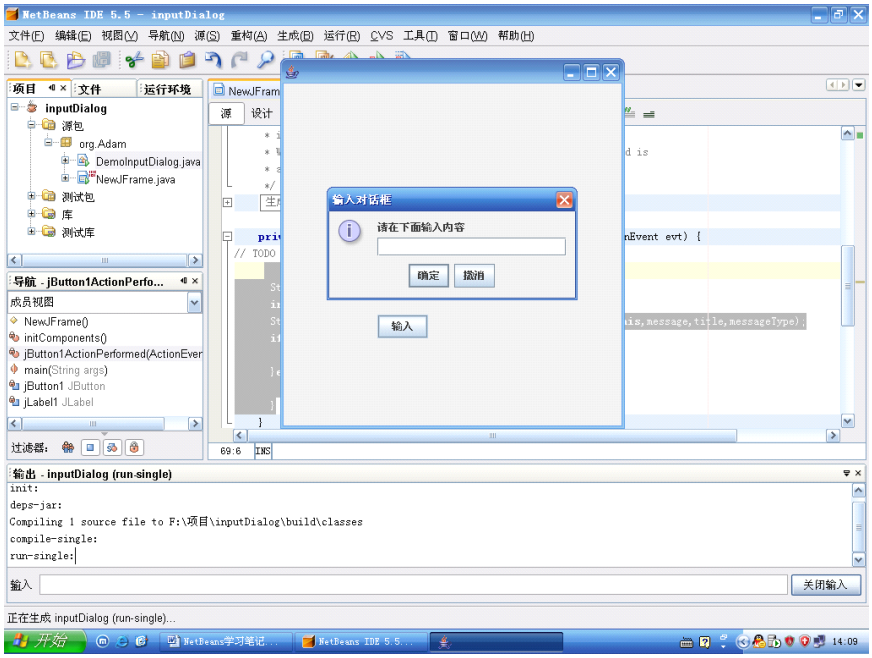


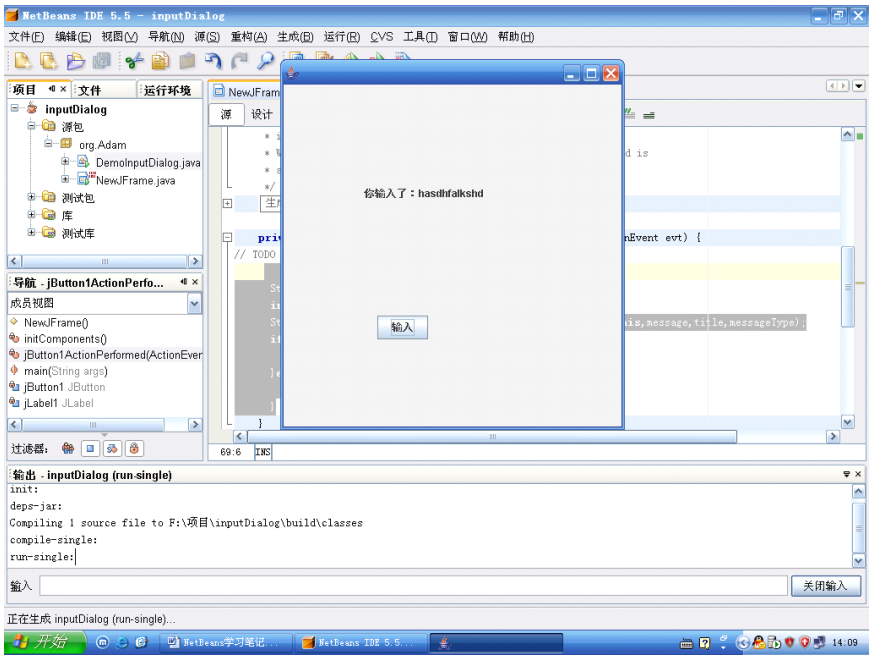

窗体上还有一个 Jlabel，在 jButton1 的事件下添加代码：

```
String title=new String("输入对话框");
    String message=new String("请在下面输入内容");
    int messageType=JOptionPane.INFORMATION_MESSAGE;
```

```
        String
inputMessage=(String)JOptionPane.showInputDialog(this,message,title,messageType);
        if(inputMessage !=null &&! inputMessage.equals("")){
            this.jLabel1.setText("你输入了："+inputMessage);
        }else{
            this.jLabel1.setText("你还没有输入任何内容！");
        }
```

效果图：



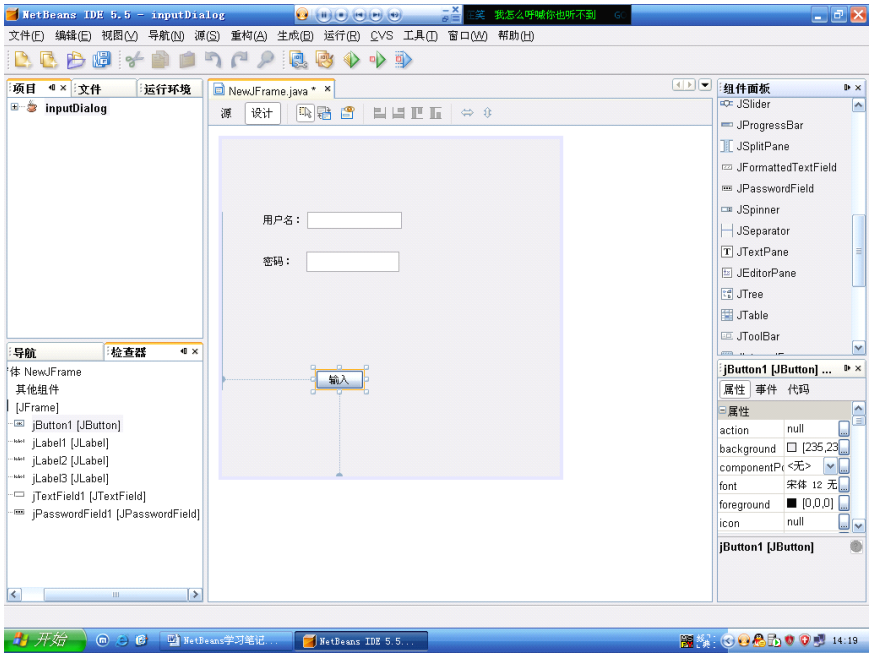




这就是输入对话框的基本使用了，你学会了吗？

如果你对对话框的使用，还是不很了解的话，让我们再来看下面的一个例子。

对于用户名和密码的判断。

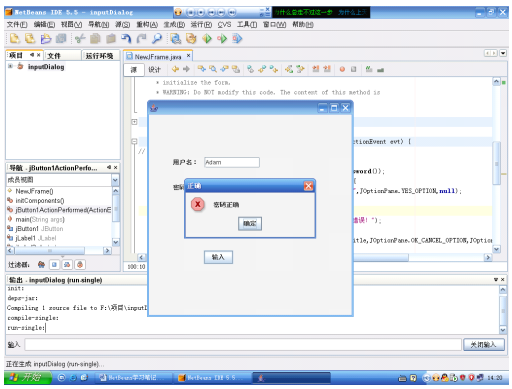
设计图：

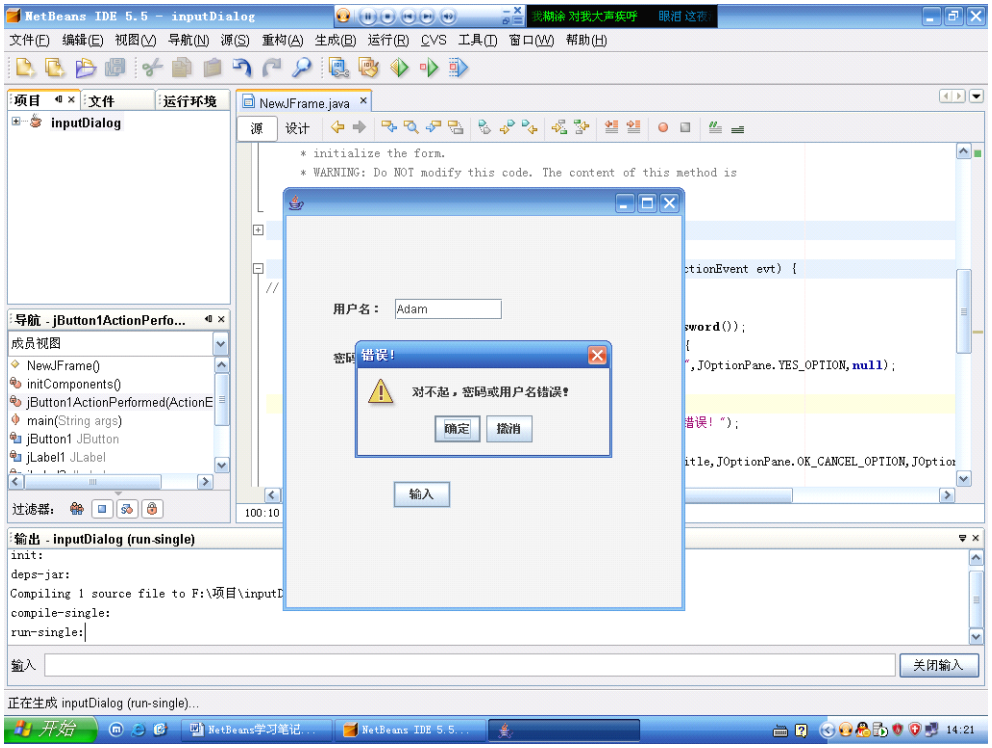



代码：

```
String name=this.jTextField1.getText();
        String Password=new String(this.jPasswordField1.getPassword());
        if( name.equals("Adam") && Password.equals("43046721")){
            JOptionPane.showMessageDialog(this," 密 码 正 确 "," 正 确 ",JOptionPane.YES_OPTION,null);
        }else{

                String message=new String("对不起，密码或用户名错误！");
                String title=new String("错误！");

JOptionPane.showConfirmDialog(this,message,title,JOptionPane.OK_CANCEL_OPTION,JOptionPane.WARNING_MESSAGE,null);
        }
```

效果图：






其实对话框的使用很简单，尤其是在 NetBeans 中，只是在我们需要使用对话框的时候，写上 JOptionPane.然后，NetBeans 自然会把相关的对话框类型显示出来，我们只是需要选择就可以了。其实，使用 NetBeans，最大的好处之一就是在于代码可以关联出来，这样就大大地节省了我们的开发时间，而且避免了一些类似于大小写的错误。

## 文件选择器（JFileChooser）

顾名思义，文件选择器就是用来打开需要的文件的。使用文件选择器，可以打开我们需要的文件。当然也可以设置一些过滤器，这样可以过滤一些我们不需要的文件类型。

文件选择器的使用非常广泛，在很多软件中都应用得到。

文件选择器的声明和使用都很简单。

最简单的声明语句：

.JFileChooser fileChooser = new JFileChooser();

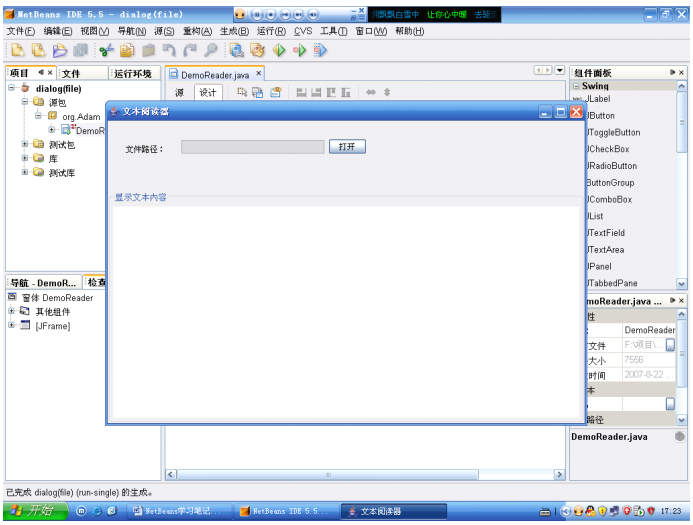
最简单的使用方法：

fileChooser.showOpenDialog(this);

下面，我们来看一个例子，这样可以帮助我们更好地了解文件选择器的使用：

效果图：

包括一个按钮，在窗体上有一个 Jpanel，在 Jpanel 上有一个 JTextArea




声明部分：

```
private javax.swing.JFileChooser fileChooser = new javax.swing.JFileChooser();
```

添加两个文件类型过滤器：

```
class TxtFliter extends javax.swing.filechooser.FileFilter {
        boolean flag;
        public boolean accept(File file){
                if(file.getName().toLowerCase().endsWith(".txt")){
                        flag=true;
                }else if(file.isDirectory()){
                        flag=true;
                }else{
                        flag=false;
                }
                 return flag;
        }
        public String getDescription(){
                return "TXT 文件";
        }
}



class JavaFliter extends javax.swing.filechooser.FileFilter {
        boolean flag;
        public boolean accept(File file){
                if(file.getName().toLowerCase().endsWith(".java")){
                        flag=true;
                }else if(file.isDirectory()){
                        flag=true;
```

```
            }else{
                flag=false;
            }
             return flag;
        }
        public String getDescription(){
            return "Java 文件";
        }
    }


    public void initFileChooser() {
            FileFilter [] fileFilter=fileChooser.getChoosableFileFilters();
            int fSize=fileFilter.length;
            for(int i=0;i<fSize;i++){
                fileChooser.removeChoosableFileFilter(fileFilter[i]);
            }
            fileChooser.addChoosableFileFilter(new TxtFliter());
            fileChooser.addChoosableFileFilter(new JavaFliter());
        }
```

在程序的构造函数中添加：

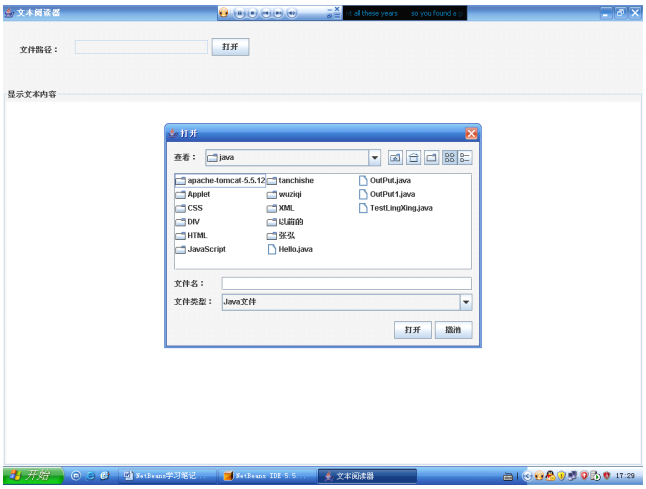
```
this.initFileChooser();
```

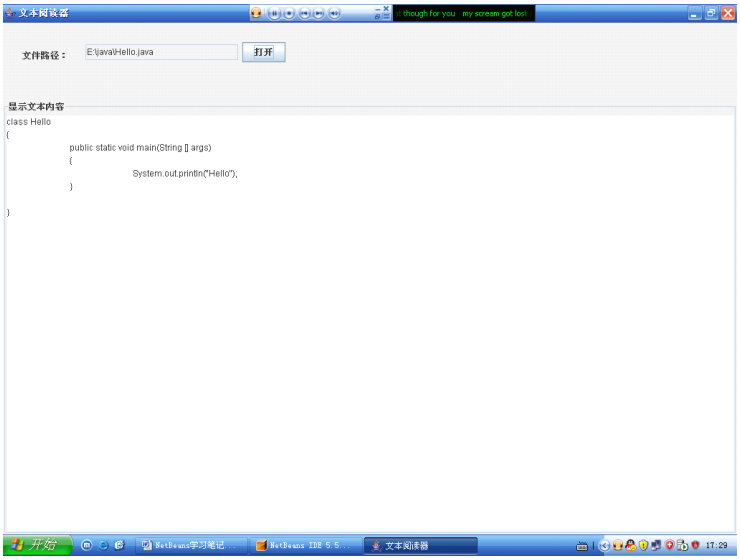
在按钮的事件中添加：

```
fileChooser.showOpenDialog(this);
            File selectFile=fileChooser.getSelectedFile();
            if(selectFile==null){
                JOptionPane.showMessageDialog(this,"你没有选择要打开的文件!","提示：
",JOptionPane.INFORMATION_MESSAGE);
                return;
            }else{
                int choose=JOptionPane.showConfirmDialog(this,"你确定要打开文件吗？
","请确认",JOptionPane.YES_NO_OPTION,JOptionPane.QUESTION_MESSAGE);
                if(choose==JOptionPane.NO_OPTION){
                    JOptionPane.showMessageDialog(this,"你取消了打开文件","提示：
",JOptionPane.INFORMATION_MESSAGE);
                    return;
                }else{
                    String tempString="";
                    BufferedReader buffRead=null;
                    String pathName=selectFile.getPath();
                    this.jTextFieldFilePath.setText(pathName);
                    try{
```

```
                int size=(int)selectFile.length();
                byte[] tempArray=new byte[size];
                FileInputStream fin=new FileInputStream(selectFile);
                fin.read(tempArray);
               tempString=new String(tempArray);
                this.jTextAreaContent.setText(tempString);

            }catch(IOException e){
                e.printStackTrace();
            }
        }
    }
```

运行效果图：



# 第 9 章 Swing 菜单

Swing 菜单是我们经常用到的一种控件，NetBeans 对菜单进行了很好的封装，是我们应用起来非常方便。下面，我们就来简单的了解一下 Swing 菜单。

菜单：Jmenu

菜单项：JmenuItem

复选菜单项：JcheckBocMenuItem

单选菜单项：JradioButtonMenuItem

弹出式菜单：JpopupMenu

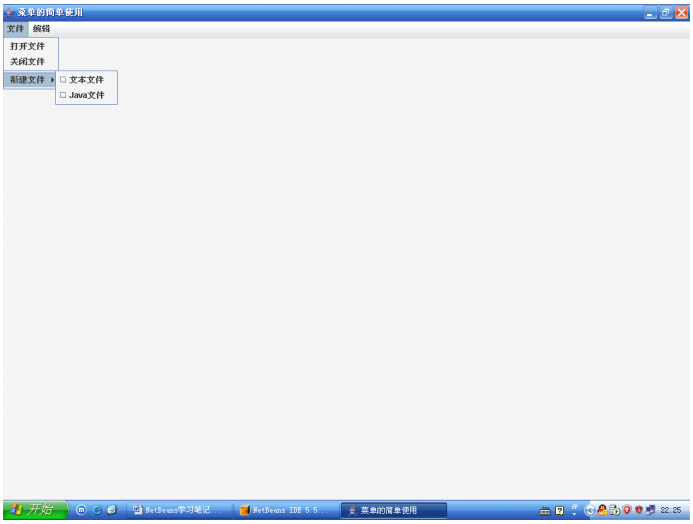
下面来看一个简单的菜单程序：

代码：

```
import java.awt.*;
import javax.swing.*;
public class DemoSimpleMenu extends JFrame{
    JMenuBar jMenuBarOne;
    JMenu fileMenu,editMenu,newFileMenu;
    JMenuItem openFile,closeFile;
    JCheckBoxMenuItem newFileTxt,newFileJava;
    JRadioButtonMenuItem copyFile,pasteFile;
    ButtonGroup buttonEdit;

    /** Creates a new instance of DemoSimpleMenu */
    public DemoSimpleMenu() {
        jMenuBarOne=new JMenuBar();
        buttonEdit=new ButtonGroup();
        fileMenu=new JMenu("文件");
        editMenu=new JMenu("编辑");
        newFileMenu=new JMenu("新建文件");
        openFile=new JMenuItem("打开文件");
        closeFile=new JMenuItem("关闭文件");
        newFileTxt=new JCheckBoxMenuItem("文本文件");
        newFileJava=new JCheckBoxMenuItem("Java 文件");
        copyFile=new JRadioButtonMenuItem("复制");
        pasteFile=new JRadioButtonMenuItem("粘贴");
        buttonEdit.add(copyFile);
        buttonEdit.add(pasteFile);
        this.setJMenuBar(jMenuBarOne);
        jMenuBarOne.add(fileMenu);
        jMenuBarOne.add(editMenu);
        fileMenu.add(openFile);
        fileMenu.add(closeFile);
        fileMenu.addSeparator();
        fileMenu.add(newFileMenu);
```

```
            newFileMenu.add(newFileTxt);
            newFileMenu.add(newFileJava);
            editMenu.add(copyFile);
            editMenu.add(pasteFile);
            this.setTitle("菜单的简单使用");
            this.setBounds(100,100,250,150);
            this.setVisible(true);
            this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        }
        public static void main(String[] args) {
            new DemoSimpleMenu();
        }
    }
```

运行图：

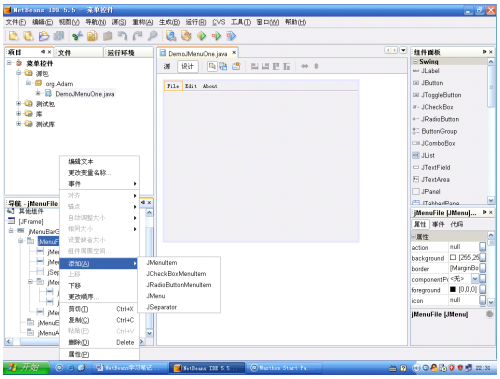



好的，现在大家已经能够明白菜单的层次结构了，那么，我们下面再看一个例子。关于菜单的各种方法，在此不作多说，在相关的帮助文档上，都可以很清楚的查到。

在 NetBeans 中使用菜单控件

在添加菜单栏，然后添加菜单，然后在导航窗口右击菜单，添加，选择要添加的菜单项，或者子菜单。

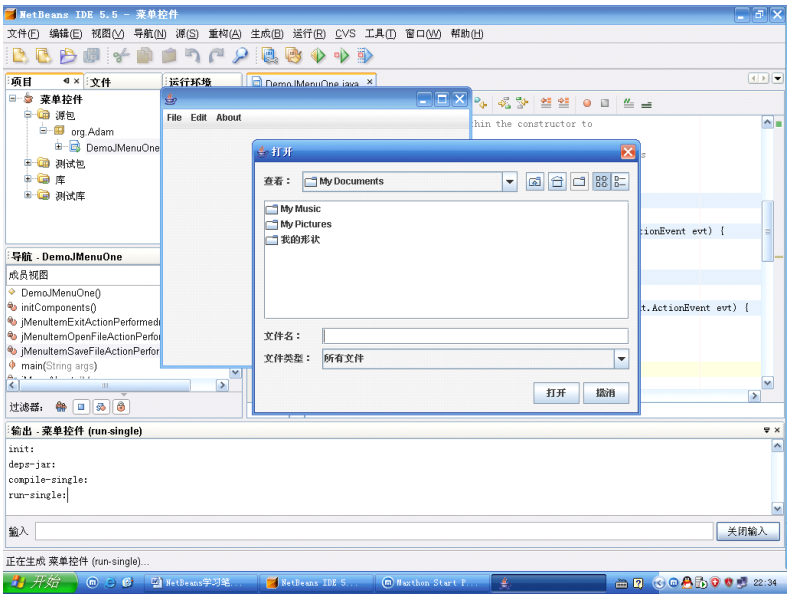
如下图：



然后在相应的菜单项上右击，可以选择事件，添加如下代码：

```
JFileChooser fileChooser=new JFileChooser();
        fileChooser.showOpenDialog(this);//打开文件菜单项

JFileChooser fileChooser=new JFileChooser();
        fileChooser.showSaveDialog(this);//保存文件菜单项
```

效果图：



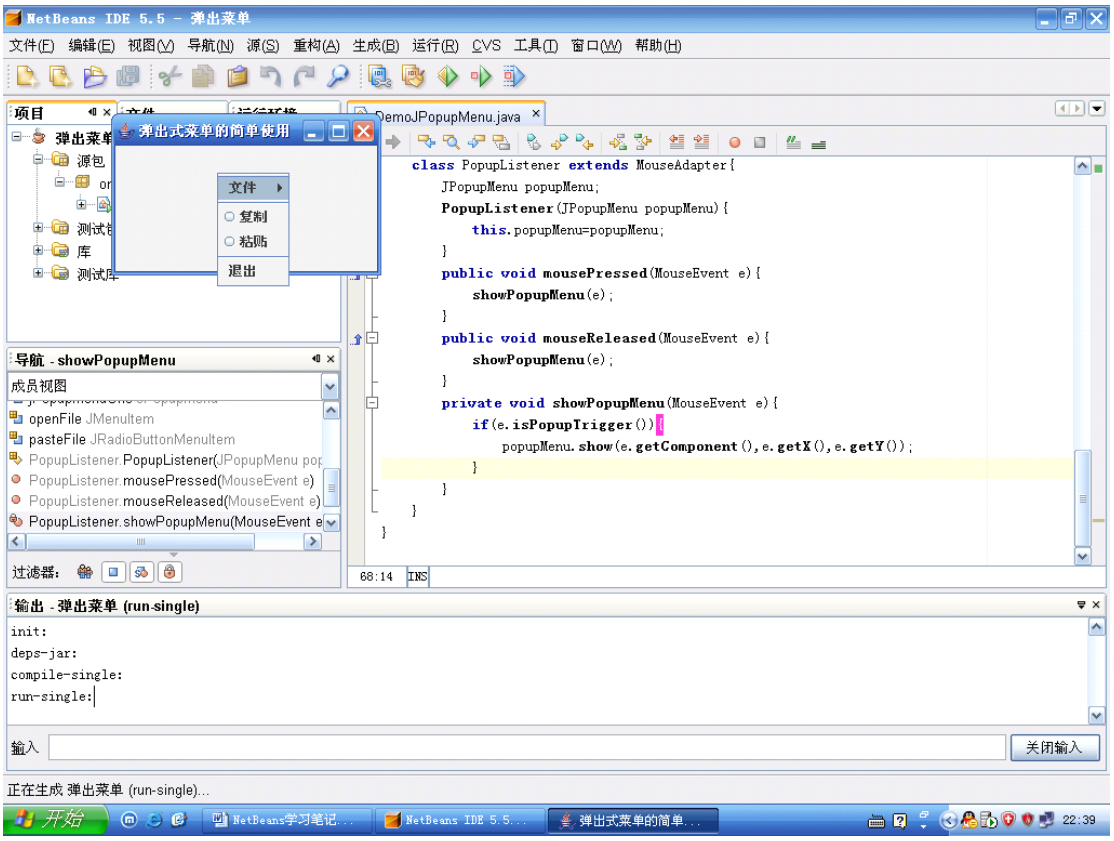

这样，我们就基本上掌握了菜单控件的开发方法，下面，我们要来讲解一个关于弹出式菜单的例子：

代码：

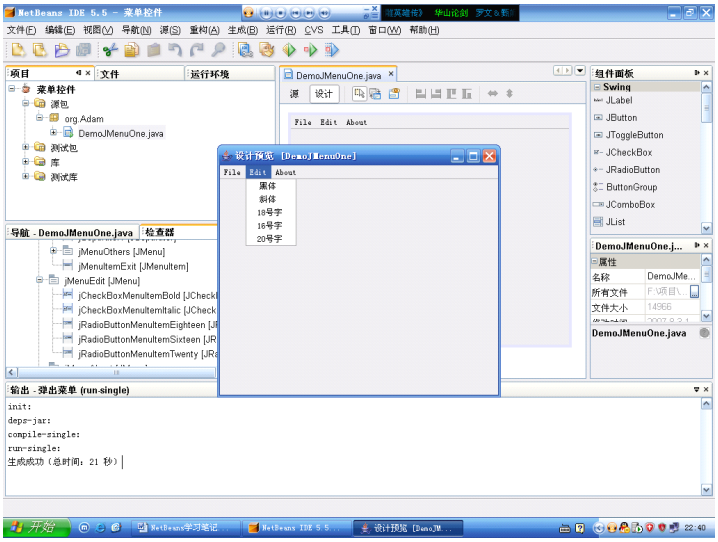
```
public class DemoJPopupMenu extends JFrame{
    JMenu fileMenu;
    JPopupMenu jPopupMenuOne;
    JMenuItem openFile,closeFile,exit;
    JRadioButtonMenuItem copyFile,pasteFile;
    ButtonGroup buttonGroupOne;
    /** Creates a new instance of DemoJPopupMenu */
    public DemoJPopupMenu() {
    jPopupMenuOne=new JPopupMenu();
    buttonGroupOne=new ButtonGroup();
    fileMenu=new JMenu("文件");
    openFile=new JMenuItem("打开");
    closeFile=new JMenuItem("关闭");
    fileMenu.add(openFile);
    fileMenu.add(closeFile);
    jPopupMenuOne.add(fileMenu);
    jPopupMenuOne.addSeparator();
    copyFile=new JRadioButtonMenuItem("复制");
```

```
        pasteFile=new JRadioButtonMenuItem("粘贴");
        buttonGroupOne.add(copyFile);
        buttonGroupOne.add(pasteFile);
        jPopupMenuOne.add(copyFile);
        jPopupMenuOne.add(pasteFile);
        jPopupMenuOne.addSeparator();
        exit=new JMenuItem("退出");
        jPopupMenuOne.add(exit);
        MouseListener popupListener=new PopupListener(jPopupMenuOne);
        this.addMouseListener(popupListener);
        this.setTitle("弹出式菜单的简单使用");
        this.setBounds(100,100,250,150);
        this.setVisible(true);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        }
        public static void main(String[] args) {
            new DemoJPopupMenu();
        }
        class PopupListener extends MouseAdapter{
            JPopupMenu popupMenu;
            PopupListener(JPopupMenu popupMenu){
                this.popupMenu=popupMenu;
            }
            public void mousePressed(MouseEvent e){
                showPopupMenu(e);
            }
            public void mouseReleased(MouseEvent e){
                showPopupMenu(e);
            }
            private void showPopupMenu(MouseEvent e){
                if(e.isPopupTrigger()){
                    popupMenu.show(e.getComponent(),e.getX(),e.getY());
                }
            }
        }
}
```

效果图：




下面，让我们最后再来看一个复选菜单项和单选菜单项的例子：




在每一个菜单项下的 StateChanged 下：

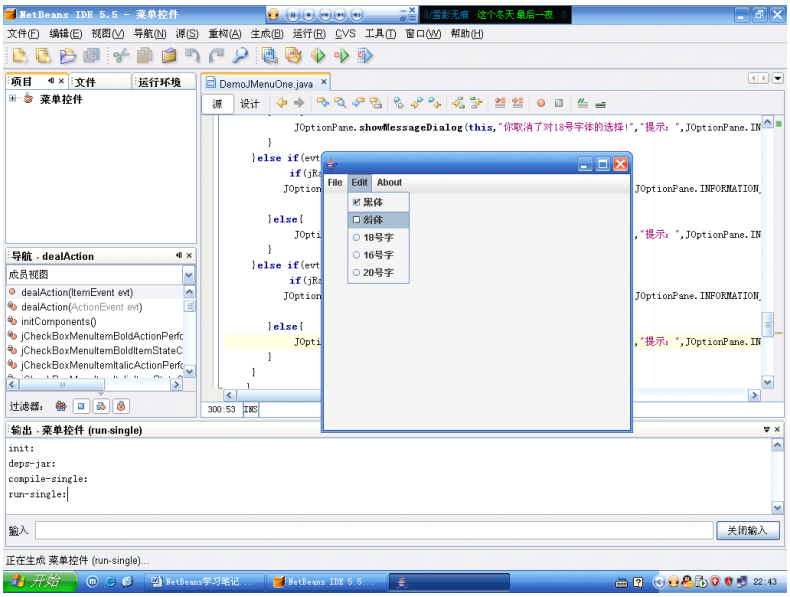
this.dealAction(evt);

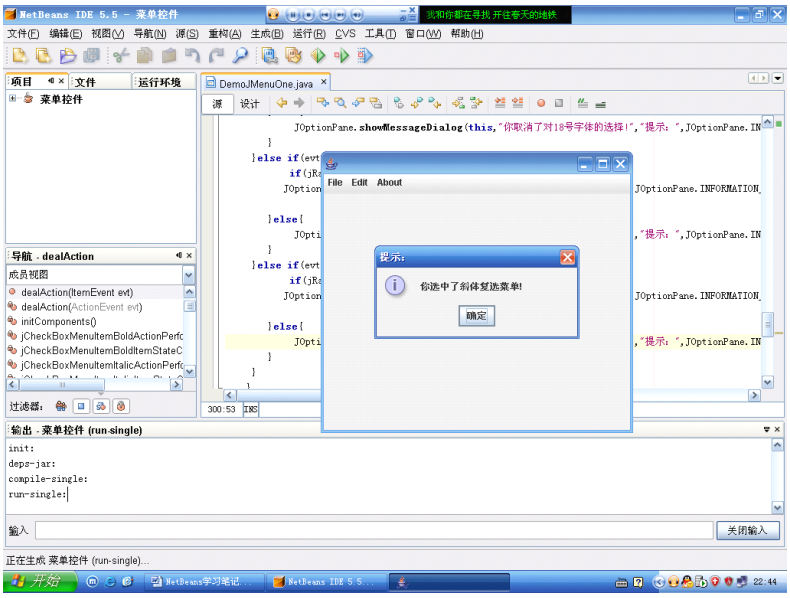
最后：

```
public void dealAction(java.awt.event.ItemEvent evt) {
        if(evt.getSource()==jCheckBoxMenuItemBold){
            if(jCheckBoxMenuItemBold.isSelected()){
            JOptionPane.showMessageDialog(this,"你选中了黑体复选菜单!","提示：
",JOptionPane.INFORMATION_MESSAGE);
```

```
                }else{
                    JOptionPane.showMessageDialog(this,"你取消了对黑体复选菜单的选
择!","提示：",JOptionPane.INFORMATION_MESSAGE);
                }
            }else if(evt.getSource()==jCheckBoxMenuItemItalic){
                if(jCheckBoxMenuItemItalic.isSelected()){
                JOptionPane.showMessageDialog(this,"你选中了斜体复选菜单!","提示：
",JOptionPane.INFORMATION_MESSAGE);

                }else{
                    JOptionPane.showMessageDialog(this,"你取消了对斜体复选菜单的选
择!","提示：",JOptionPane.INFORMATION_MESSAGE);
                }
            }else if(evt.getSource()==jRadioButtonMenuItemEighteen){
                 if(jRadioButtonMenuItemEighteen.isSelected()){
                JOptionPane.showMessageDialog(this,"你选中了 18 号字体!","提示：
",JOptionPane.INFORMATION_MESSAGE);

                }else{
                    JOptionPane.showMessageDialog(this,"你取消了对 18 号字体的选择!","提
示：",JOptionPane.INFORMATION_MESSAGE);
                }
            }else if(evt.getSource()==jRadioButtonMenuItemSixteen){
                 if(jRadioButtonMenuItemSixteen.isSelected()){
                JOptionPane.showMessageDialog(this,"你选中了 16 号字体!","提示：
",JOptionPane.INFORMATION_MESSAGE);

                }else{
                    JOptionPane.showMessageDialog(this,"你取消了对 16 号字体的选择!","提
示：",JOptionPane.INFORMATION_MESSAGE);
                }
            }else if(evt.getSource()==jRadioButtonMenuItemTwenty){
                 if(jRadioButtonMenuItemTwenty.isSelected()){
                JOptionPane.showMessageDialog(this,"你选中了 20 号字体!","提示：
",JOptionPane.INFORMATION_MESSAGE);

                }else{
                    JOptionPane.showMessageDialog(this,"你取消了对 20 号字体的选择!","提
示：",JOptionPane.INFORMATION_MESSAGE);
                }
            }
        }
```

效果图：





关于图像

图像类 Image

一些常见的方法 ：

getImage(java.net.URL)　通过指定的地址获取图像对象

例 ：

Jframe jf=new Jframe();

Image ii=jf.getToolkit().getImage(“C;\\hello.gif”);

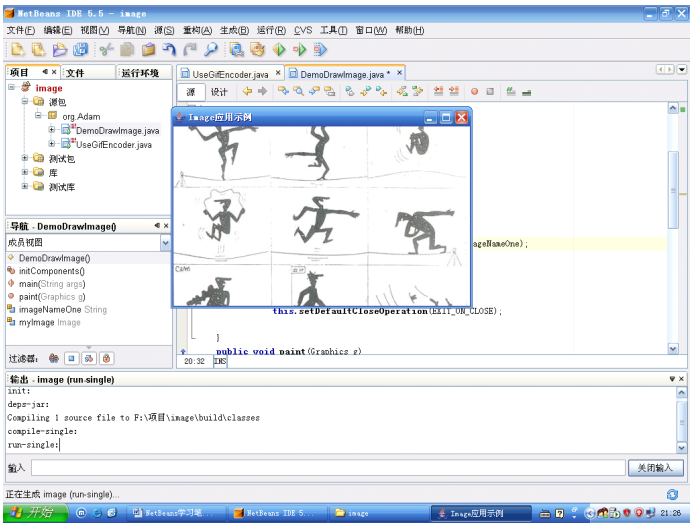
其实，在程序开发的过程中，图像对我们来说十分重要，可以更好的美化我们的界面。一个成功的软件，其界面的美化程度，是十分重要的。

在上面的例子中，我们曾经举过一个关于窗口图标的例子，那就是图像在程序开发中应用的一个很好的例子。

下面，我们再来看几个例子：
源代码：

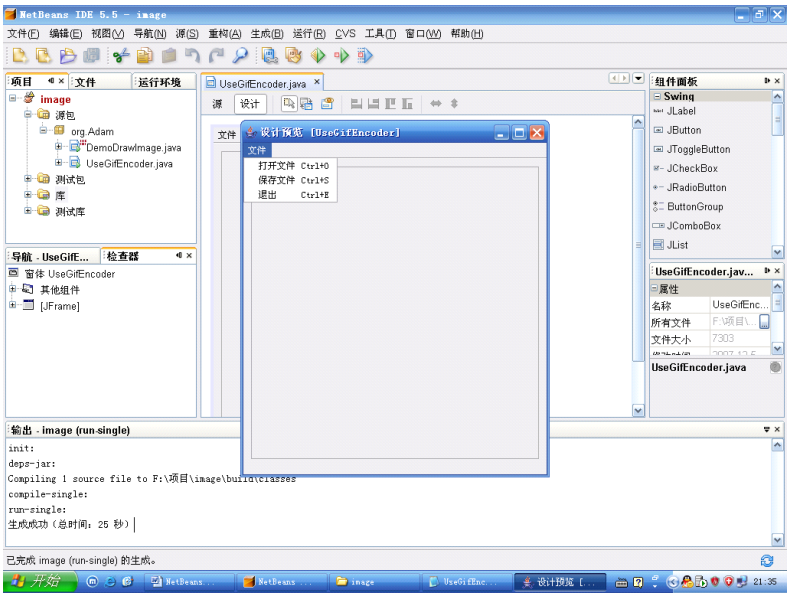
```
import java.awt.*;
import javax.swing.*;

public class DemoDrawImage extends JFrame
{
	Image myImage;
	String imageNameOne=new String("happy.jpg");
	public DemoDrawImage()
	{
		myImage=Toolkit.getDefaultToolkit().getImage(imageNameOne);
		this.setBounds(250,150,450,300);
		this.setTitle("Image 应用示例");
		this.validate();
		this.setVisible(true);
		this.setDefaultCloseOperation(EXIT_ON_CLOSE);

	}
	public void paint(Graphics g)
	{
		g.drawImage(myImage,0,0,this);
		//g.dispose();
	}
	public static void main(String args[])
	{
		new DemoDrawImage();
	}
}
```

效果图：



通过上面的这个例子，我们就可以学会怎样获得图像了。

下面，我们来看一个例子，学习怎样使用 GifEncoder 编码器的程序的功能开发
创建一个窗体，添加菜单，还有菜单项，具体见下图：




窗体上还要有一个很大的 Jlabel，来存放图片。
下面是源代码：

```
private GifEncoder encoder ;


    private JFileChooser fileChooser = new JFileChooser();

    private File selectFile = null;

    private File openedFile;

    private void openImageFile() {
        fileChooser.showOpenDialog(this);
        openedFile=fileChooser.getSelectedFile();
        if(openedFile==null)
            return;
        else{
            this.jLabelImage.setIcon(new ImageIcon(openedFile.getPath()));

        }
    }

    public void saveImageAsGif() {
        try{
            int saved=fileChooser.showSaveDialog(this);
```

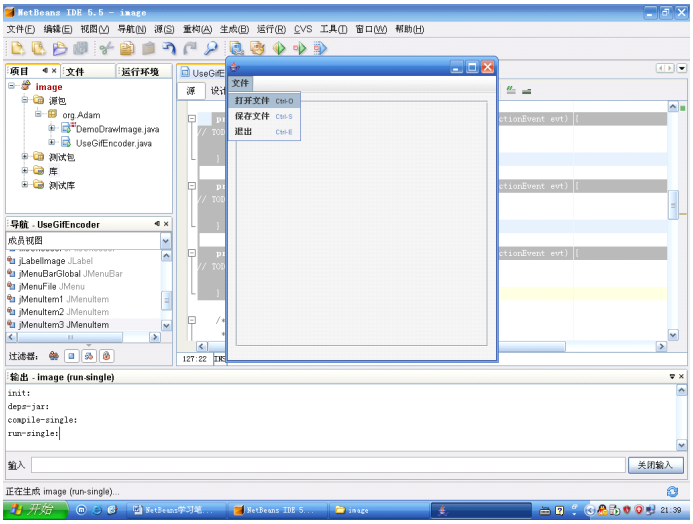
```
                if(saved==JFileChooser.APPROVE_OPTION){
                    if(this.openedFile !=null){
                        File saveFileName=fileChooser.getSelectedFile();
                        String fileName=new String(saveFileName.getPath()+".gif");
                        FileOutputStream fileOutPut=new FileOutputStream(fileName);
                        InputStream ins=new FileInputStream(this.openedFile.toString());
                        BufferedImage srcImage=ImageIO.read(ins);
                        encoder = new GifEncoder(srcImage, fileOutPut);
                        encoder.encode();
                        fileOutPut.flush();
                        fileOutPut.close();
                        ins.close();
                    }
                }

            }catch(Exception ea){
                ea.printStackTrace();
            }


        }
```

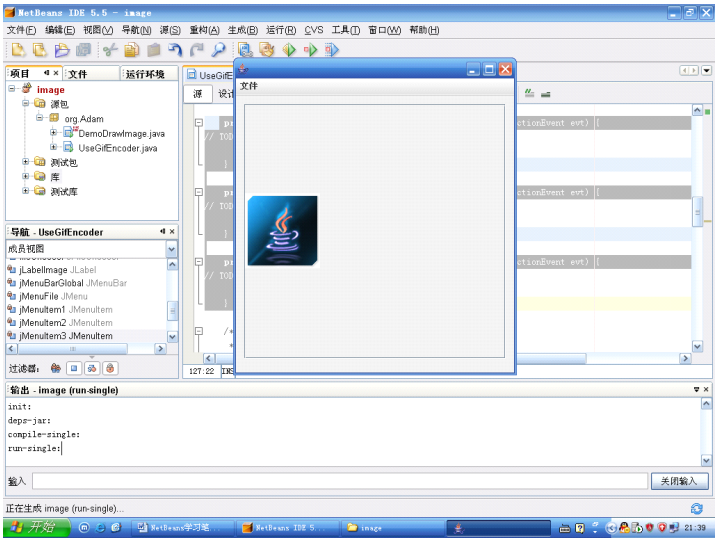
在相应的菜单项下添加相应的代码：

```
    private void jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO 将在此处添加您的处理代码：
            System.exit(0);
        }

        private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO 将在此处添加您的处理代码：
            this.saveImageAsGif();
        }

        private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO 将在此处添加您的处理代码：
            this.openImageFile();
        }
```

在这个程序开发的最重要的环节，就是在“库”的位置上右击，添加 JAR 文件，然后选择路径，添加上 openmap.jar 的文件。

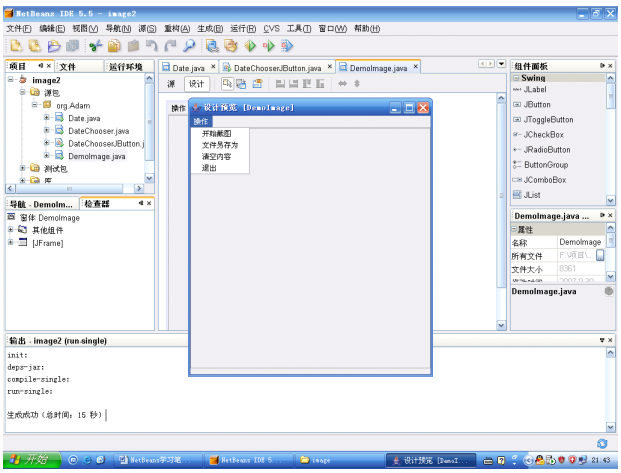
效果图：





那么，通过上面的学习，我们已经对 java 中的图像处理有了一定的了解了，但是这些还不是很足够，要想很好的应用，必须进行充分的练习。那么，下面，我们再来看最后一个关于图像处理的例子：截图工具的开发。

部署图：



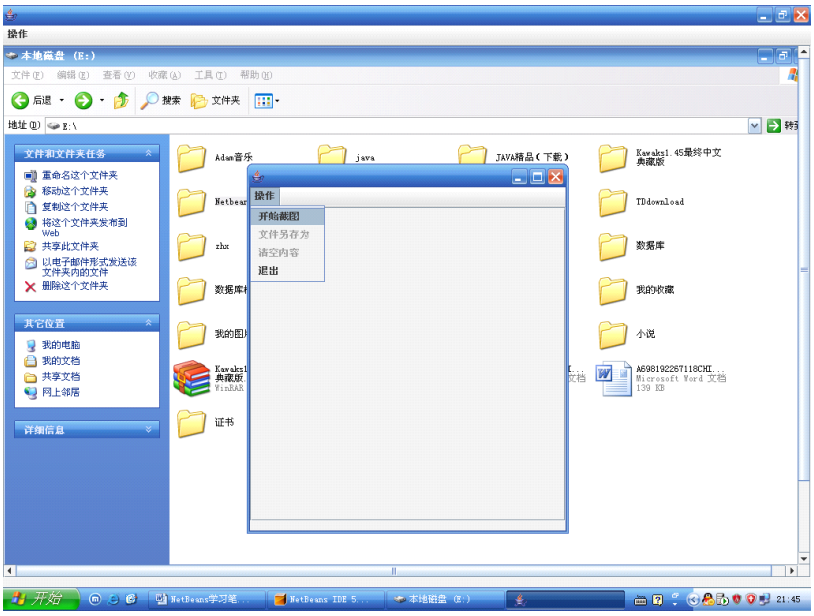
源代码：

```
private Image createImage() {
            try {//截图代码开始
                  Image tempLocalImage=null;
                  Robot robot=new Robot();
                  Dimension dimension= Toolkit.getDefaultToolkit().getScreenSize();
                  Rectangle scrRect=new Rectangle(0,0,dimension.width,dimension.height);
                  tempLocalImage=robot.createScreenCapture(scrRect);
                   //截图代码结束
                   return tempLocalImage;
            } catch (AWTException ex) {
                  ex.printStackTrace();
            }
            return null;
      }
private void jMenuItem4ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO 将在此处添加您的处理代码：
            System.exit(0);
      }

      private void jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO 将在此处添加您的处理代码：
            this.jLabelShowIcon.setIcon(null);
            this.jMenuItem2.setEnabled(false);
            this.jMenuItem3.setEnabled(false);
            this.jMenuItem1.setEnabled(true);
      }

      private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO 将在此处添加您的处理代码：
             this.saveImage();
      }

      private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO 将在此处添加您的处理代码：
            jLabelShowIcon.setIcon(new ImageIcon(tempImage=this.createImage()));
           this.jMenuItem2.setEnabled(true);
           this.jMenuItem3.setEnabled(true);
           this.jMenuItem1.setEnabled(false);
      }
      public void saveImage()
      {
             try
      {
```

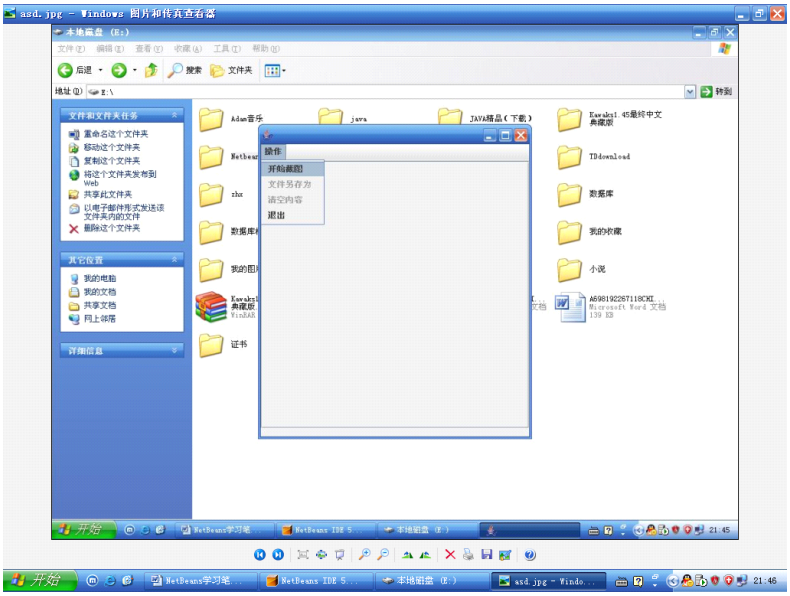
```
                int saved=fileChooser.showSaveDialog(this);
                if(saved==JFileChooser.APPROVE_OPTION)
                {
                    //获取要保存
                    File saveFileName=fileChooser.getSelectedFile();
                    //获取要保存文件的名字
                    String fileName=new String(saveFileName.getPath()+".jpg");
                    //1.首先创建一个输出流
                    FileOutputStream fileOutPut=new FileOutputStream(fileName);
                   //2.然后把输出流用 JPEG 编码器进行包裹，其实就是把输出流连接到编
码器
                    encoder=JPEGCodec.createJPEGEncoder(fileOutPut);
                    //把 BufferedImage 对象进行编码
                    encoder.encode((BufferedImage) tempImage);
                    fileOutPut.flush();
              fileOutPut.close();
                    }
                }
          catch(Exception ea)
          {
                ea.printStackTrace();
          }
          }
    private Image tempImage;
    private JPEGImageEncoder encoder;
    private JFileChooser fileChooser=new JFileChooser();
```

效果图：

这是点击开始解图后的效果



点击文件另存为：




这是截下来的图。